40 DE JOCURI LOGICE IN BASIC



ION DIAMANDI GHEORGHE PĂUN

40 de JOCURI LOGICE în BASIC

CUPRINS

PREFAȚĂ

Despre calculatoare în general și despre calculatoarele de tip « personal » în particular se vorbește în ultimul timp cu tot mai multă insistență, această minunată invenție a secolului XX fiind pe cale să provoace modificări de mare profunzime în viața noastră. Incepînd cu activitatea cotidiană (petrecerea timpului liber, documentare, activitate profesională etc.) și terminînd cu conducerea proceselor economice, cu un accent deosebit în educație, unde se așteaptă o adevărată revoluție cauzată de (bazată pe) informatică. Pe de o parte, știința nouă a informaticii își reclamă un loc printre disciplinele de învățămînt, un loc pe care îl merită ținînd seama de importanța ei economico-socială și pe care îl și obține treptat-treptat, începînd cu învățămîntul superior și coborînd spre cel liceal, gimnazial și chiar mai jos. Pe de altă parte, calculatorul poate fi folosit cu mare succes ca mijloc de învățămînt, ca material didactic inteligent, făcînd un pas esențial dincolo de materialele tradiționale. Deosebit de atractiv, cu o comportare « umană » din multe puncte de vedere, simulînd adică o comportare inteligentă, cu posibilitatea importantă a feedback-ului, a dialogului, înglobînd efecte vizuale și sonore « animate », de o mare fiabilitate — iată doar cîteva dintre calitățile care fac din calculator un ajutor de mare valoare al profesorului și un tovarăș de mare eficiență al elevului.

Apare aici un aspect extrem de important, care ușurează enorm atît accesul la informatică (cel puțin la anumite niveluri ale acesteia), cît și implicarea calculatorului în învățare: spiritul ludic ce animă primii (mulți) ani ai dezvoltării personalității umane, dar care nu este străin omului la nici o vîrstă, și posibilitățile ludice ale calculatorului, în special ale celor personale. « Jucărie infinită și inteligentă », aceasta este una din « definițiile » cele mai potrivite pentru calculatoarele personale. Și răspîndirea lor

deosebită din ultimii ani tot pe asemenea «aplicaţii» se bazează. **A intra în informatică prin joc, a învăţa matematică, fizică, chimie, limbi străine prin joc, acesta este idealul oricărui copil (şi educator).** *Plus plăcerea însăşi a jocului, chiar dacă nu este vorba despre învăţarea a ceva anume (în mod explicit), fiorul competiţiei, fie ea de natură logică sau bazată pe îndemînare, pe decizii rapide, exacte, atracţia aventurii, în ipostaza aproape cinematografică pe care ne-o oferă ecranul controlat de cele 30—40 de butoane. Efectul aparent este proliferarea extraordinară a jocurilor programate;* pentru calculatoarele din familia **Sinclair ZX Spectrum** *(incluzînd, deci, calculatoarele româneşti compatibile,* **TIM-S, HC-85, COBRA, CIP-01, JET),** există cîteva mii de asemenea jocuri în circulaţie. *Şi, desigur, apar mereu jocuri noi. În 1988, RECOOP a lansat pe piaţă prima casetă cu jocuri pe calculator, ajungînd astăzi la caseta cu nr. 10.*

Mai mult, RECOOP a lansat pe piaţă sub emblema «JECO», cîteva sute de jocuri logice educative — colective, care se pretează a fi realizate şi pe calculator.

Informatica în acest domeniu aduce, însă, un avantaj de posibilităţi inedite, deci, de jocuri inedite, care vor suscita interesul, uneori chiar amplificat, din partea tinerei generaţii.

Lucrarea de faţă se înscrie exact în această direcţie de manifestare a (micro)informaticii. Ea vă propune 40 de jocuri, programate în BASIC, limbajul cel mai răspîndit pentru calculatoarele personale. Ele sînt funcţionale pentru calculatoarele din familia **Sinclair ZX Spectrum.** *Pentru a facilita adaptarea (transcrierea) acestor jocuri pentru alte calculatoare (aMIC, PRAE, TPD JUNIOR, FELIX PC, CIP-01 etc.), s-a căutat să se reducă la minimum liniile de instrucţiuni care fac apel la particularităţi de construcţie ale calculatoarelor din familia* **Sinclair ZX Spectrum** *(de exemplu, instrucţiuni POKE, cu care se modifică conţinutul unor locaţii de memorie).*

Sînt alese jocuri cît mai diferite, ca tip şi ca acţiune propriu-zisă, jocuri logice competitive (la care un partener este calculatorul), jocuri solitare, multe jocuri de reflexe; este inclus şi un program (cu o putere de joc considerabilă) pentru popularul joc «şeptică», programe (jocuri) didactice, utile în formarea deprinderilor de calcul numeric, pentru învăţarea sub formă de joc a unor noţiuni de matematică, biologie, chimie gimnazială etc., precum şi două jocuri de «aventură»: dezlegarea unei enigme pentru găsirea unei comori într-o peşteră şi un program în care trebuie luate cele mai bune decizii pentru a ieşi nevătămat din hăţişurile junglei.

În afară de încercarea de scriere cît mai compactă, programele nu au folosit idei de realizare deosebite, principalele două obiective avute în vedere fiind claritatea şi . . . buna funcţionare. Tot pentru a facilita înţelegerea lor, mai ales în cazul în care cititorul doreşte să opereze modificări (pentru unele jocuri au fost sugerate unele modificări şi în lucrare), pentru majori-

tatea programelor au fost date explicaţii detaliate, pe grupuri de instrucţiuni. Din acest motiv şi-din cel al economiei de spaţiu, s-a renunţat, în general, la liniile REM, de comentariu.

Pentru a uşura urmărirea programelor, în listinguri caracterele grafice sînt identificate prin includerea literelor-taste respective între semnele "< >". Dacă obţinerea caracterului grafic presupune şi acţionarea tastei CAPS SHIFT, atunci, după semnul "<", va apărea şi indicaţia CAPS. Exemple: PRINT "< ab >" — după obţinerea modului grafic, utilizatorul va apăsa tastele A şi B; PRINT "< CAPS 8888 >" — după obţinerea modului grafic, utilizatorul va ţine apăsată tasta CAPS SHIFT şi va acţiona de patru ori tasta 8.

O serie de «efecte speciale», tehnici şi artificii de programare în BASIC au fost prezentate la sfîrşitul lucrării, pentru cei care doresc să perfecţioneze grafica şi «coloana sonoră» a programelor, eventual să protejeze anumite linii ale acestora. Multe programe pot fi îmbunătăţite şi ca algoritm propriu-zis, nu neapărat ca putere de joc, ci ca eficienţă — timp de răspuns sau memorie folosită (s-a semnalat acest lucru, de exemplu, în cazul programului Masteract*); de obicei, aici am «sacrificat» criteriul* memorie *pentru a cîştiga timp (a se vedea, de pildă, cazul jocurilor «de evitare», la care am reţinut într-o matrice formaţiile de evitat — pătrate, triplete — posibile pe întreaga tablă, pentru a nu le mai identifica în cursul desfăşurării partidei).*

Aşa cum este concepută, lucrarea doreşte, pe de o parte, să ofere, pur şi simplu, un număr de jocuri celor pasionaţi de calculatoare, pe de altă parte, să dea posibilitatea perfecţionării în programare, în utilizarea limbajului BASIC (aprofundînd, de pildă, cunoştinţele dobîndite în urma parcurgerii lucrării, «Partenerul meu de joc . . . calculatorul»*, publicată de RECOOP în 1988, aflată acum la ediţia a doua). În plus, prin jocurile incluse, se oferă un generos «teren de antrenament» pentru inteligenţă, pentru îndemînare, pentru spiritul de competiţie, pentru formarea gîndirii algoritmice (şi chiar pentru fixarea unor cunoştinţe de matematică, chimie, biologie). Rămîne ca cititorul să confirme sau să infirme atingerea acestor obiective.*

AUTORII

tatea programelor au fost date explicaţii detaliate, pe grupuri de instrucţiuni. Din acest motiv şi din cel al economiei de spaţiu, s-a renunţat, în general, la liniile REM, de comentariu.

Pentru a uşura urmărirea programelor, în listinguri caracterele grafice sînt identificate prin includerea literelor-taste respective între semnele < >. Dacă obţinerea caracterului grafic presupune şi acţionarea tastei CAPS SHIFT, atunci, după semnul <, va apărea şi indicaţia CAPS. Exemple: PRINT "< ab > — după obţinerea modului grafic, utilizatorul va apăsa tastele A şi B; PRINT "< CAPS 8888 > — după obţinerea modului grafic, utilizatorul va ţine apăsată tasta CAPS SHIFT şi va acţiona de patru ori tasta 8.

O serie de « efecte speciale », tehnici şi artificii de programare în BASIC au fost prezentate la sfîrşitul lucrării, pentru cei care doresc să perfecţioneze grafica şi « coloana sonoră » a programelor, eventual să protejeze anumite linii ale acestora. Multe programe pot fi îmbunătăţite şi ca algoritm propriu-zis, nu neapărat ca putere de joc, ci ca eficienţă — timp de răspuns sau memorie folosită (s-a semnalat acest lucru, de exemplu, în cazul programului Masteract); de obicei, aici am « sacrificat » criteriul memorie pentru a cîştiga timp (a se vedea, de pildă, cazul jocurilor « de evitare », la care am reţinut într-o matrice formaţiile de evitat — pătrate, triplete — posibile pe întreaga tablă, pentru a nu le mai identifica în cursul desfăşurării partidei).

Aşa cum este concepută, lucrarea doreşte, pe de o parte, să ofere, pur şi simplu, un număr de jocuri celor pasionaţi de calculatoare, pe de altă parte, să dea posibilitatea perfecţionării în programare, în utilizarea limbajului BASIC (aprofundînd, de pildă, cunoştinţele dobîndite în urma parcurgerii lucrării, « Partenerul meu de joc . . . calculatorul », publicată de RECOOP în 1988, aflată acum la ediţia a doua). În plus, prin jocurile incluse, se oferă un generos « teren de antrenament » pentru inteligenţă, pentru îndemînare, pentru spiritul de competiţie, pentru formarea gîndirii algoritmice (şi chiar pentru fixarea unor cunoştinţe de matematică, chimie, biologie). Rămîne ca cititorul să confirme sau să infirme atingerea acestor obiective.

AUTORII

Jocuri logice competitive

În general, un joc este numit « logic » atunci cînd el se adresează inteligenței, cînd mutările (în sens larg, ca acțiuni ale jucătorilor) se fac după dorință și după un plan liber ales, fără intervenția hazardului. Matematicienii numesc aceste jocuri « cu informație completă »: în fiecare moment există o mulțime (de obicei finită) de mutări posibile, cunoscute de toți participanții la întrecere și alegerea între o mutare sau alta o fac jucătorii, fără restricții exterioare. Un joc este « competitiv » atunci cînd la el participă două sau mai multe persoane/echipe, fiecare avînd dorința și posibilitatea de a cîștiga. Evident, într-un joc « serios », numai unul dintre competitori cîștigă. La unele jocuri, există și posibilitatea de remiză, rezultat nedecis. Modul de a defini victoria (și remiza, dacă există) diferă de la un joc la altul, dar există o serie de categorii mari de finaluri posibile prin: capturarea unei anumite piese a adversarului **(Șah)**, capturarea tuturor pieselor adversarului sau a marii lor majorități **(Dame, Moară)**, realizarea unei anumite configurații a pieselor **(Cinci în rînd)**, mutarea pieselor dintr-o poziție în alta **(Halma)**, deplasarea unei piese într-un anume cîmp **(Tablut)**, aducerea adversarului în situația de a nu mai putea muta **(Vulpea și gîștele)**, evaluarea unui punctaj de diferite tipuri **(GO, Reversi)**. În unele jocuri, nu învingătorul ci învinsul este mai ușor de definit; este cazul jocurilor « de evitare », în care cel care realizează cu piesele sale o anumită configurație pierde.

O caracteristică generală a jocurilor logice competitive este că fiecare mutare trebuie aleasă conform unui plan, bazat, de obicei, pe evaluarea consecințelor mutării la pașii următori ai partidei. Aceasta face ca în cele mai multe dintre cazuri să fie nevoie de o permanentă analiză a evoluției jocului, de o explorare cît mai avansată a arborelui de evoluție. În cazul programării (pentru a pune calculatorul să joace — eventual bine), acest lucru dă complicații serioase, ca efort de programare, memorie necesară și timp de răspuns la fiecare mutare. Există însă și jocuri la care, din analiza unei configurații a pieselor (static, deci), se poate deduce care este mutarea (cea mai bună) de efectuat. De asemenea, există jocuri la care simpla grijă de a nu greși grav conduce la mutări rezonabile pentru că . . . adversarul poate greși mai mult. Desigur, jocurile din ultimele două categorii sînt mult mai ușor de programat și mai ales asupra unor jocuri de acest tip ne-am oprit în această lucrare.

Unele programe au o tărie de joc derizorie (asta nu înseamnă că pot fi subestimate), cele mai multe joacă rezonabil, iar cîteva sînt imbatabile (jocuri care au strategie de cîştig cunoscută, iar programul aplică o asemenea strategie). Care este tipul fiecărui program, veţi descoperi singuri (în prezentarea jocurilor am specificat de obicei acest lucru). Indiferent însă de puterea unui program, confruntarea cu el este benefică, jocurile logice fiind totdeauna un teren ideal de antrenare a inteligenţei, a puterii de analiză, a memoriei, a spiritului de competitivitate, de ordine. Iar cînd partener este calculatorul, pe de o parte, rigoarea trebuie suplimentată (în unele programe, o mutare greşit tastată nu mai poate fi modificată), pe de altă parte, vi se oferă un vast cîmp de manifestare a propriilor idei strategice, pe care le puteţi adăuga programelor, încercînd să le măriţi puterea de joc (atunci cînd acest lucru este posibil). Foloasele sînt de fiecare dată evidente.

TIC TAC TOE

Jocul este foarte vechi (unele variante erau cunoscute și de romani) și destul de răspîndit, mai ales în rîndul copiilor (care îl numesc «X și zero»). Pe o tablă caroiată, de dimensiuni 3×3, doi jucători (unul cu piese albe, celălalt cu piese negre) așază pe rînd cîte o piesă proprie, într-un cîmp liber. Cel care realizează un șir de trei piese proprii, orizontal, vertical sau diagonal, cîștigă partida. Dacă tabla este acoperită fără ca partida să fie cîștigată de un jucător, se consideră remiză. În program, cele nouă cîmpuri ale tablei de joc sînt numerotate ca în figura 1.

Fig. 1

1	2	3
4	5	6
7	8	9

Jocul este destul de simplu. El a fost bine analizat și se știe că dacă ambii jucători mută corect, partida se încheie totdeauna remiză. Este și cazul programului de față, imbatabil (dar care știe să cîștige atunci cînd adversarul greșește).

La începutul unei partide, programul întreabă «Cine joacă primul (C=calculatorul/J=jucătorul)?», iar la sfîrșit — dacă se dorește «Alt joc (d/n)?». Mutările jucătorului sînt indicate prin precizarea cîmpului unde se dorește așezarea unei piese (1—9). Calculatorul joacă cu piese albe, jucătorul cu negrele.

Descrierea programului

30—45 — se desenează tabla de joc.

100 — variabila **mut** numără mutările (dacă nu se cîștigă mai devreme, la **mut** = 9 se oprește jocul).

120 — se colorează ecranul.

160 — se marchează cîmpurile (matricea **c** conține coordonatele acestora pe ecran).

170—200 — opțiunea de începere a jocului.

220 — **juc** indică jucătorul aflat la mutare (**juc** = 1 pentru calculator, **juc** = 2 pentru jucător).

230 (GOSUB 1100) — se efectuează mutarea: în matricea **a** se înregistrează cine a mutat în acel cîmp, iar pe ecran apare piesa respectivă; piesele sînt desenate la liniile 1000—1040 și reținute în vectorul **a$** (piesa albă pe prima poziție — «a» în modul grafic — și cea neagră pe a doua poziție — «b» în modul grafic); în matricea **c** sînt memorate coordonatele celor nouă cîmpuri ale tablei de joc pe ecran (liniile 1080—1085).

260 (GOSUB 900) — se așteaptă mutarea jucătorului.

280—340 — se examinează prima mutare a jucătorului și se execută dacă este corectă.

350 — la început, calculatorul mută în funcție de valoarea lui **q** (alături de prima

mutare a jucătorului sau pe diagonală, după cum acesta a jucat pe o latură sau într-un colț al tablei).
370 — **ind** = 1 indică faptul că deja calculatorul a cîştigat.
390—450 — mutările ulterioare ale jucătorului.
460 (GOSUB 1500) — subrutina 1500 alege mutarea calculatorului; în matricea **d** (liniile 1060—1065) sînt memorate toate tripletele de pe tablă, iar între 1500—1560 se testează dacă se poate cîştiga într-o mutare sau dacă adversarul cîştigă într-o mutare (pentru a fi blocat); în matricea **e** (liniile 1070—1075) sînt memorate toate dublele amenințări de pe tablă; liniile 1570—1630 caută să joace duble amenințări pentru a cîştiga la pasul următor; dacă nu există, se joacă orice loc gol (liniile 1633—1670), într-un cîmp care are asociat un număr par dacă se poate (liniile 1640—1645).
470—530 — « sărbătorirea » sonoră a victoriei calculatorului şi reluarea jocului.
600—680 — dacă jucătorul mută primul şi o face în centru, calculatorul răspunde în colț (cîmpul 1).
710—750 — dacă apoi jucătorul mută în cîmpul 9, calculatorul joacă în 3.
790 — numai primele trei mutări ale partidei sînt tratate individual, după aceea se foloseşte subrutina 1500.

```
   2 CLS : PRINT AT 3,6; FLASH 1
;"    TIC-TAC-TOE     "
   5 GO SUB 1000
  10 CLS : BORDER 1
  20 PRINT AT 3,6; FLASH 1;"
TIC-TAC-TOE "
  30 FOR i=1 TO 4
  35 PLOT 64+24*i,56: DRAW 0,72
  40 PLOT 88,32+24*i: DRAW 72,0
  45 NEXT i
  50 LET ind=0
 100 DIM a(9): LET mut=0
 110 OVER 1: PAPER 7: BORDER 1
 120 FOR i=0 TO 21: FOR j=0 TO 2
8 STEP 4: PRINT INK 2;AT i,j;"(C
APS 8888)": BEEP .003,i+j: NEXT
j: NEXT i
 130 OVER 0
 160 FOR i=1 TO 9: PRINT AT c(i,
1)-1,c(i,2)+2;i: NEXT i
 170 PRINT AT 21,0;"Cine muta pr
imul (C=calc/J=juc)?"
 180 PAUSE 0: LET r$=INKEY$: BEE
P .1,22
 190 IF r$="c" THEN GO TO 220
 200 IF r$="j" THEN GO TO 600
 210 BEEP 1,-6: GO TO 170
 220 LET loc=5: LET juc=1
 230 GO SUB 1100
 260 GO SUB 900
 280 IF r$="2" OR r$="8" THEN LE
T q=1: GO TO 320
 290 IF r$="4" OR r$="6" THEN LE
T q=3: GO TO 320
 300 IF r$="1" OR r$="3" OR r$="
7" OR r$="9" THEN LET q=-10: GO
TO 320
 310 BEEP 1,-6: GO TO 260
 320 LET loc=VAL r$: LET juc=2
 330 IF loc=5 THEN GO TO 310
 340 GO SUB 1100
```

```
 350 LET loc=ABS (loc+q): LET ju
c=1
 360 GO SUB 1100
 370 IF ind=1 THEN GO TO 470
 380 IF mut=9 THEN GO TO 540
 390 GO SUB 900
 410 IF r$>="1" AND r$<="9" THEN
GO TO 430
 420 BEEP 1,-6: GO TO 390
 430 LET loc=VAL r$: LET juc=2
 440 IF a(loc)<>0 THEN GO TO 420
 450 GO SUB 1100: IF mut=9 THEN
GO TO 540
 460 GO SUB 1500: LET juc=1: GO
TO 360
 470 FOR i=1 TO 5: FOR j=7 TO 1
STEP -1
 480 BORDER j: BEEP .02*i,i*j
 490 NEXT j: NEXT i
 500 PRINT AT 18,1;"  AM CISTIGA
T !!! "
 510 PAUSE 60: PRINT AT 21,1;"Al
t joc (d/n) ?"
 520 PAUSE 0: IF INKEY$<>"d" THE
N STOP
 530 GO TO 10
 540 PRINT AT 18,2;" REMIZA "
 550 BEEP .2,22: BEEP .5,15: BEE
P 1,22
 560 GO TO 510
 600 GO SUB 900
 620 IF r$="5" THEN GO TO 650
 630 IF r$>="1" AND r$<="9" THEN
GO TO 800
 640 BEEP 1,-6: GO TO 600
 650 LET loc=5: LET juc=2: GO SU
B 1100
 680 LET loc=1: LET juc=1: GO SU
B 1100
 690 GO SUB 900
```

```
 710 IF r$="9" THEN GO TO 740
 720 IF r$>="1" AND r$<="9" THEN
GO TO 770
 730 BEEP 1,-6: GO TO 690
 740 LET loc=9: LET juc=2: GO SU
B 1100
 750 LET loc=3: LET juc=1: GO SU
B 1100
 760 GO TO 390
 770 LET loc=VAL r$: LET juc=2:
GO SUB 1100
 780 LET loc=10-loc: LET juc=1:
GO SUB 1100
 790 GO TO 390
 800 LET loc=VAL r$: LET juc=2:
GO SUB 1100
 810 LET loc=5: LET juc=1: GO SU
B 1100
 820 GO TO 390
 900 PRINT AT 21,0;"     Astept m
utarea ta (1 - 9) "
 910 PAUSE 0: LET r$=INKEY$: BEE
P .1,22
 920 RETURN
 999 STOP
1000 FOR i=0 TO 7: READ x: POKE
USR "a"+i,x: NEXT i
1010 DATA 60,66,129,129,129,129,
66,60
1020 FOR i=0 TO 7: READ x: POKE
USR "b"+i,x: NEXT i
1030 DATA 60,126,255,255,255,255
,126,60
1040 DIM a$(2): LET a$(1)="<a>":
 LET a$(2)="<b>"
1050 DIM d(20,3): DIM e(16,5): D
IM c(9,2)
1060 FOR i=1 TO 20: FOR j=1 TO 3
: READ d(i,j): NEXT j: NEXT i
1065 DATA 1,2,3,1,5,9,1,4,7,2,1,
3,2,5,8,3,1,2,3,5,7,3,6,9,4,1,7,
4,5,6,6,4,5,6,3,9,7,1,4,7,5,3,7,
8,9,8,7,9,8,2,5,9,1,5,9,3,6,9,7,
8
1070 FOR i=1 TO 16: FOR j=1 TO 5
: READ e(i,j): NEXT j: NEXT i
1075 DATA 1,2,3,5,9,1,2,3,4,7,1,
5,9,4,7,2,1,3,5,8,3,1,2,5,7,3,1,
2,6,9,3,5,7,6,9,4,1,7,5,6,6,4,5,
3,9,7,1,4,5,3,7,1,4,8,9,7,5,3,8,
9,8,7,9,2,5,9,1,5,3,6,9,1,5,7,8,
9,3,6,7,8
1080 FOR i=1 TO 9: READ c(i,1):
READ c(i,2): NEXT i
1085 DATA 8,10,8,13,8,16,11,10,1
1,13,11,16,14,10,14,13,14,16
1090 RETURN
1100 LET a(loc)=juc: PRINT AT 21
,0;"
   "
1110 PRINT AT c(loc,1)-1,c(loc,2
)+2; FLASH 1;a$(juc)
1120 PAUSE 60: PRINT AT c(loc,1)
-1,c(loc,2)+2;a$(juc)
1130 BEEP .1,12: BEEP .2,22
1140 LET mut=mut+1: RETURN
1500 LET ind=0: LET loc2=10
1510 FOR i=1 TO 20
1520 LET d1=d(i,1): LET d2=d(i,2
): LET d3=d(i,3)
1530 IF a(d1)=0 AND a(d2)=1 AND
a(d3)=1 THEN LET loc=d1: LET ind
=1: RETURN
1540 IF a(d1)=0 AND a(d2)=2 AND
a(d3)=2 THEN LET loc2=d1
1550 NEXT i: LET t=0
1560 IF loc2<>10 THEN LET loc=lo
c2: RETURN
1570 FOR i=1 TO 16
1580 LET e1=e(i,1): LET e2=e(i,2
): LET e3=e(i,3): LET e4=e(i,4):
 LET e5=e(i,5)
1590 IF a(e1)<>0 THEN GO TO 1630
1600 LET sum1=a(e2)+a(e3)
1605 LET sum2=a(e4)+a(e5)
1610 IF sum1=1 AND sum2=1 THEN L
ET loc=e1: RETURN
1620 IF sum1=2 AND sum2=2 THEN L
ET loc2=e1: LET t=t+1
1630 NEXT i
1633 IF t=0 THEN GO TO 1650
1635 IF t=1 THEN LET loc=loc2: R
ETURN
1640 FOR i=2 TO 8 STEP 2: IF a(i
)=0 THEN LET loc=i: RETURN
1645 NEXT i
1650 FOR i=9 TO 1 STEP -1
1660 IF a(i)=0 THEN LET loc=i: R
ETURN
1670 NEXT i: RETURN
```

NIM

Jocul este foarte vechi (variante erau cunoscute încă în China antică) și este complet « rezolvat » din punct de vedere matematic. Varianta din acest program (o formă deghizată apare și în **Duelul broscuţelor**) este următoarea: se dau 1—9 șiruri de obiecte identice, în fiecare șir găsindu-se cîte 1—25 obiecte. Numărul de șiruri este stabilit de jucător; de asemenea, el poate spune și cîte obiecte se găsesc în fiecare șir sau poate să lase programul să facă acest lucru (la întîmplare, folosind generatorul de numere aleatoare). Pe rînd, jucătorul și calculatorul iau dintr-un șir arbitrar oricîte obiecte doresc, desigur, măcar unu. Nu pot fi luate la aceeași mutare obiecte din două șiruri diferite. Cel care nu mai poate juca (toate șirurile au fost epuizate) pierde partida. Indiferent cine stabilește șirurile de plecare, programul întreabă « Cine joacă primul (C = calculatorul / J = jucătorul)? ».
Această posibilitate face ca programul să poată fi învins ușor, alegînd, de exemplu, un singur șir, jucînd primul și luînd toate obiectele din șir. Într-o partidă «cinstită», programul joacă însă **Nim** perfect, de aceea el este greu de învins. Iar strategia de urmat (poate fi găsită în numeroase lucrări de jocuri matematice) este relativ simplă: se scriu în baza 2 numerele obiectelor din fiecare șir (de exemplu, dacă avem trei șiruri, cu cîte 7, 24 și, respectiv, 9 obiecte, obținem 7 = 00111, 24=11000, 9=01001), apoi însumăm aceste numere **fără transport** spre stînga, pentru 1+1 = 10 nereținînd adică decît cifra 0, nu și cifra 1, de ordin superior. În exemplul dinainte, obținem:

```
 00111+
 11000
 01001
 -----
 10110
```

Dacă în suma obținută astfel există cifre 1 (este cazul anterior), atunci jucătorul aflat la mutare în acel moment poate cîștiga. Invers, dacă nu apar decît cifre 0 (și dacă adversarul cunoaște strategia cîștigătoare . . .), atunci jucătorul aflat la mutare va pierde. Regula de mutare care asigură cîștigul este următoarea: se iau obiecte în așa fel încît suma să fie formată numai din cifre zero. Pentru aceasta, se alege o linie care are o cifră 1 pe coloana celui mai din stînga 1 din sumă (în cazul nostru, linia a doua, cea corespunzătoare numărului 24) și se schimbă pe această linie cifrele 1 cu 0 și cifrele 0 cu 1, pe toate pozițiile unde în sumă apăreau cifre 1. Numărul în baza doi obținut astfel reprezintă numărul de obiecte care trebuie să rămînă în șirul descris de linia respectivă. În exemplul considerat mai devreme, linia a doua trebuie modificată pe pozițiile 1, 3 și 4 și ea devine 01110, ceea ce reprezintă numărul 14 în baza doi. Trebuie, deci, să ridicăm din șirul al doilea 24—14= 10 obiecte.
Pentru mutare, jucătorul este întrebat « Din ce rînd iei? » și va trebui să tasteze o cifră între 1 și 9, corespunzătoare șirului de obiecte din care dorește să mute. Dacă mutarea este posibilă (șirul nu este epuizat), atunci programul întreabă « Cîte obiecte? » și se tastează totdeauna două cifre, prima fiind zero pentru numere mai mici decît 10 (se tastează 08 pentru 8, nu simplu 8).
După orice partidă, jocul poate fi reluat (se întreabă « Alt joc (d/n)? »).

Descrierea programului

10 (GOSUB 1000) — matricea **c** conține cifrele în baza 2 ale numerelor între 0 și 25 (mărimea posibilă a grămezilor din joc); se încarcă la liniile 1010 — 1020; vectorul **m** conține puterile lui 2, pentru a fi folosite la calcularea mutărilor **Nim** cîștigătoare; se încarcă la liniile 1030—1040; obiectele din grămezi sînt desenate la liniile 1050—1060 (« a » în modul grafic — pe liniile 390 și 1120).
15 — vectorul **a** conține mărimea curentă a rîndurilor/grămezilor.
20—100 — pregătirea ecranului.
110—140 — precizarea numărului de rînduri (variabila **ri**).
150—190 — opțiunea de completare a rîndurilor.
200—290 — jucătorul propune mărimea fiecărui rînd (din cîte două cifre, prima fiind, deci, înmulțită cu 10 — linia 230).
300—335 — calculatorul propune mărimea fiecărui rînd.
340 — **sum** este numărul curent de obiecte disponibile (cînd **sum** = 0, jocul se încheie)
350—410 — desenarea rîndurilor (și calcularea variabilei **sum**).
420—460 — cine începe?
470—590 — se introduce și se verifică mutarea jucătorului.
600 (GOSUB 1100) — ștergerea de pe rîndul ales a obiectelor și diminuarea variabilei **sum**; cînd **sum** = 0, variabila **ind** = 1 indică terminarea partidei.
610—690 — a cîștigat jucătorul; opțiunea de reluare.
700—900 — căutarea unei mutări **Nim** cîștigătoare; **sb** este suma cifrelor binare de pe fiecare coloană (se calculează la liniile 720—780); **bb** = 1 indică existența unei coloane cu suma 1 (deci existența unei mutări **Nim** cîștigătoare); mutarea **Nim** cîștigătoare este calculată pe liniile 820—890.
920—970 — dacă nu există o mutare **Nim** cîștigătoare, atunci se mută din rîndul maxim, cu jumătate din el, dacă lungimea lui este mai mare de 5 (liniile 920—960) sau cu un singur pas, în caz contrar (linia 970).
980 — și mutările calculatorului sînt efectuate tot în subrutina 1100.

```
  10 BORDER 1: GO SUB 1000
  15 CLS : INK 2: PAPER 6: DIM a
(9)
  20 FOR i=0 TO 31
  30 PRINT AT 0,i;"<CAPS 8>"
  40 PRINT AT 19,i;"<CAPS 8>"
  50 PRINT AT 21,i;"<CAPS 8>"
  60 NEXT i
  70 FOR i=1 TO 20
  80 PRINT AT i,0;"<CAPS 8>"
  90 PRINT AT i,31;"<CAPS 8>"
 100 NEXT i
 105 INK 0
 110 PRINT AT 20,1;"Cite rinduri
 (1 - 9) ?"
 120 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 130 IF r$<"1" OR r$>"9" THEN BE
EP 1,-6: GO TO 120
 140 LET ri=VAL r$
 150 PRINT AT 20,1;"Cine propune
 rindurile (c/j) ?"
 160 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 170 IF r$="c" THEN GO TO 300
 180 IF r$="j" THEN GO TO 200
 190 BEEP 1,-6: GO TO 160
```

```
 200 FOR i=1 TO ri
 210 PRINT AT 20,1;"Rindul ";i;"
 =                  "
 220 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 230 IF r$>="0" AND r$<="9" THEN
 LET a(i)=10*VAL r$: GO TO 250
 240 BEEP 1,-6: GO TO 210
 250 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 260 IF r$>="0" AND r$<="9" THEN
 LET a(i)=a(i)+VAL r$: GO TO 280
 270 BEEP 1,-6: GO TO 210
 280 PRINT AT 20,12;a(i): PAUSE
50
 285 IF a(i)=0 OR a(i)>25 THEN B
EEP 1,-6: GO TO 210
 290 NEXT i: GO TO 340
 300 PRINT AT 20,1;"Bine - propu
n imediat          "
 310 FOR i=1 TO ri
 320 LET a(i)=INT (RND*25)+1
 330 NEXT i
 335 PAUSE 60
 340 LET sum=0
 350 PRINT AT 20,1;"Iata situati
a de plecare
```

```
 360 FOR i=1 TO ri
 370 PRINT AT 2*i-1,2; INVERSE 1
;i: LET sum=sum+a(i)
 380 FOR j=1 TO a(i)
 390 PRINT AT 2*i-1,3+j;"(a)": B
EEP .02,i+j+10
 400 NEXT j
 410 NEXT i
 420 PRINT AT 20,1;"Cine incepe
(C= calc/J= juc) ?"
 430 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 440 IF r$="c" THEN GO TO 700
 450 IF r$="j" THEN GO TO 470
 460 BEEP 1,-6: GO TO 430
 470 PRINT AT 20,1;"Din ce rind
vrei sa iei ?     "
 480 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 490 IF r$<"0" OR r$>"9" THEN B
EEP 1,-6: GO TO 480
 500 LET ia=VAL r$
 510 IF a(ia)=0 THEN BEEP 1,-6:
GO TO 480
 520 PRINT AT 20,1;"Cite piese i
ei din rindul ";ia;" ?"
 530 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 540 IF r$>="0" AND r$<="9" THEN
 LET mut=10*VAL r$: GO TO 560
 550 BEEP 1,-6: GO TO 530
 560 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 570 IF r$>="0" AND r$<="9" THEN
 LET mut=mut+VAL r$: GO TO 590
 580 BEEP 1,-6: GO TO 530
 590 IF mut>a(ia) THEN GO TO 580
 600 GO SUB 1100
 610 IF ind=0 THEN GO TO 700
 620 PRINT AT 20,1;"Ai cistigat
- felicitari !!  "
 630 FOR i=1 TO 5
 640 FOR j=7 TO 1 STEP -1: BORDE
R j: BEEP .1,i*j: NEXT j
 650 NEXT i
 660 PRINT AT 20,1;"Alt joc (d/n
) ?                  "
 670 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 680 IF r$="d" THEN GO TO 15
 690 STOP
 700 PRINT AT 20,1;"E rindul meu
                   "
 710 LET bb=0
 720 FOR j=1 TO 5
 730 LET sb=0
 740 FOR i=1 TO 9
 750 LET nr=a(i)+1
 760 LET sb=sb+c(nr,j)
 770 IF sb=2 THEN LET sb=0
 780 NEXT i
 790 IF sb=0 THEN GO TO 900
 800 IF bb=1 THEN GO TO 870
 810 LET bb=1
 820 FOR i=1 TO 9
 830 LET nr=a(i)+1
 840 IF c(nr,j)=0 THEN GO TO 860
 850 LET ia=i: LET mut=m(j): GO
TO 900
 860 NEXT i
 870 LET nr=a(ia)+1
 880 IF c(nr,j)=0 THEN LET mut=m
ut-m(j): GO TO 900
 890 LET mut=mut+m(j)
 900 NEXT j
 910 IF bb=1 THEN GO TO 980
 920 LET max=0
 930 FOR i=1 TO 9
 940 IF a(i)>max THEN LET max=a(
i): LET ia=i
 950 NEXT i
 960 IF max>=5 THEN LET mut=INT
(max/2): GO TO 980
 970 LET mut=1
 980 GO SUB 1100: IF ind=0 THEN
GO TO 470
 990 PRINT AT 20,1;"Am cistigat
!!!                 ": GO TO 630
1000 DIM c(26,5): DIM m(5)
1010 FOR i=1 TO 26: FOR j=1 TO 5
: READ c(i,j): NEXT j: NEXT i
1020 DATA 0,0,0,0,0,0,0,0,0,1,0,
0,0,1,0,0,0,0,1,1,0,0,1,0,0,0,0,
1,0,1,0,0,1,1,0,0,0,1,1,1,0,1,0,
0,0,0,1,0,0,1,0,1,0,1,0,0,1,0,1,
1,0,1,1,0,0,0,1,1,0,1,0,1,1,1,0,
0,1,1,1,1,1,0,0,0,0,1,0,0,0,1,1,
0,0,1,0,1,0,0,1,1,1,0,1,0,0,1,0,
1,0,1,1,0,1,1,0,1,0,1,1,1,1,0,
0,0,1,1,0,0,1
1030 FOR i=1 TO 5: READ m(i): NE
XT i
1040 DATA 16,8,4,2,1
1050 FOR i=0 TO 7: READ x: POKE
USR "a"+i,x: NEXT i
1060 DATA BIN 00011100,BIN 00111
110,BIN 01100111,BIN 01111111,BI
N 01101111,BIN 00101110,BIN 0001
1100,0
1070 RETURN
1100 LET ind=0
1110 FOR j=a(ia)-mut+1 TO a(ia)
1120 PRINT AT 2*ia-1,j+3; FLASH
1;"(a)": BEEP .03,12+i
1130 NEXT j
1135 PAUSE 60
1140 FOR j=a(ia)-mut+1 TO a(ia)
1150 PRINT AT 2*ia-1,j+3;" ": BE
EP .03,32+i
1160 NEXT j
1170 LET a(ia)=a(ia)-mut
1180 LET sum=sum-mut
1190 IF sum=0 THEN LET ind=1
1200 RETURN
```

duelul broscuțelor

Jocul nu este altceva decît un **Nim** deghizat, cu poziția de plecare fixată, de aceea i se aplică întreaga teorie a jocului **Nim**. Concret, avem 10 « piste », la capetele cărora sînt așezate, față în față, cîte o broscuță. Broscuțele din stînga pot fi mutate spre dreapta, pe pista pe care se găsesc, cu un număr arbitrar de pași; cele din dreapta sînt mutate de calculator spre stînga. Broscuțele nu pot sări unele peste altele, deci, atunci cînd ajung față în față, se blochează și pe pista respectivă nu se mai pot face deplasări. Cel care nu mai poate muta (toate pistele au fost « consumate », broscuțele sînt față în față) pierde partida.
Este evident că avem un joc **Nim** cu 10 grămezi, obiectele de ridicat fiind pașii înainte pe care îi fac broscuțele. La începutul partidei, cele 10 piste conțin cîte 7, 9, 19, 19, 17, 15, 9, 7, 15, 15 locuri goale.
Scriind aceste numere în baza doi, obținem:

```
 7—00111
 9—01001
19—10011
19—10011
17—10001
15—01111
 9—01001
 7—00111
15—01111
15—01111
   -----
   11110
```

prin urmare, primul jucător care mută cîștigă (făcînd 6 pași pe pista 3 sau 4, ori 2 pași pe pista 5). Onest, programul întreabă însă la început « Cine joacă primul (C = calculatorul/J = jucătorul)? », oferindu-vă, deci, o șansă.
Pentru mutarea unei broscuțe, se alege mai întîi pista, deplasînd în jos — cu ajutorul tastei 6 — semnul indicator triunghiular, care se găsește în stînga pistelor. Apoi, apăsînd tasta 8, broscuța de pe rîndul ales se mută cu un pas spre dreapta. Cînd deplasarea se consideră terminată (și s-a făcut măcar un pas, altfel mutarea nu este acceptată), se apasă tasta 9 și trece la mutare calculatorul. Acesta, desigur, joacă **Nim** perfect; de aceea, dacă nu jucați conform « teoriei », vă va învinge aproape sigur. Dacă jucați primul și urmați strategia de la **Nim**, atunci calculatorul va juca oarecum la întîmplare (din rîndul cel mai lung, fie cu jumătate din rînd, fie numai un pas etc.).
La terminarea unei partide există posibilitatea reluării sale (se pune întrebarea « Alt joc (d/n)? »).

Descrierea programului

30—110 — se desenează marginile ecranului.
120—170 — se caroiază ecranul.
210—250 — se desenează broscuțele, la capetele pistelor; coordonatele capetelor pistelor sînt memorate în matricea **a**, încărcată la liniile 1010, 1020; ulterior, matricea **a** va conține coordonatele curente ale broscuțelor; broscuțele sînt desenate la liniile 1090—1120 (cea din stînga este identificată prin « a » în modul grafic, cea din dreapta prin « b » în modul grafic).
260 — **sum** este numărul de pași care pot fi efectuați la un moment dat (la început, **sum** = 132).

270—310 — cine începe?

320 — variabila ind indică dacă s-a făcut deja o deplasare a unei broscuțe sau nu; **în caz afirmativ ind = 1), indicatorul** triunghiular — desenat la liniile 1130—1140 și identificat prin «c» în modul grafic — nu mai poate fi deplasat.

325 — **s** este pista pe care se găsește indicatorul triunghiular.

330—380 — comanda jucătorului.

390—420 — se mută indicatorul triunghiular.

430 — vectorul **b** conține lungimea curentă a pistelor (numărul de pași care se mai pot face pe fiecare); se încarcă la liniile 1030—1040.

440—470 — deplasarea unei broscuțe spre dreapta (cu modificarea corespunzătoare a matricelor **a** și **b** — linia 460).

480 — terminarea deplasării.

490—540 — victoria jucătorului **(sum** = 0, deci, nu se mai pot face deplasări) și reluare.

560—730 — se caută o mutare **Nim** cîștigătoare; matricea **c** (încărcată la liniile 1050—1060) conține cifrele scrierii în baza 2 a tuturor numerelor de la 0 la 20 (valorile posibile ale componentelor vectorului **b**); **bb** = 1 indică existența unei mutări cîștigătoare; **sb** este suma cifrelor binare pe o coloană (este calculat pentru fiecare dintre cele cinci coloane posibile, pe liniile 570—620); numărul de pași pe care îi va face broscuța este reținut în variabila **mut** și se calculează la liniile 645—720 în cazul existenței unei mutări **Nim** cîștigătoare; vectorul **m** (încărcat la liniile 1070—1080) conține puterile lui 2 corespunzătoare coloanelor pe care se modifică cifre în momentul căutării mutării.

740—800 — dacă nu există mutări **Nim** cîștigătoare (pe toate coloanele avem zero la sumă — deci **bb** = 0), atunci fie se mută cu jumătate din pista cea mai lungă, dacă această pistă are cel puțin 5 pași (liniile 750—790), fie cu un singur pas, în caz contrar (linia 800).

810—900 — se efectuează mutarea calculatorului și se verifică terminarea partidei (linia 880).

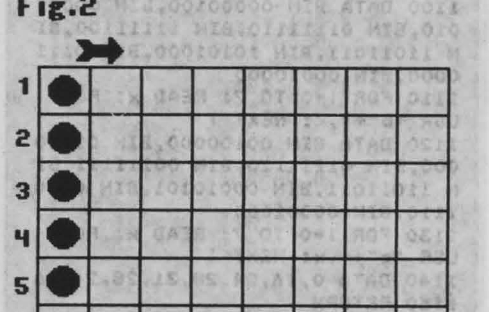
```
 10 CLS : RESTORE : GO SUB 1000
 20 INK 0: BORDER 1: PAPER 6
 30 FOR i=0 TO 31
 40 PRINT AT 0,i;"<CAPS 8>"
 50 PRINT AT 19,i;"<CAPS 8>"
 60 PRINT AT 21,i;"<CAPS 8>"
 70 NEXT i
 80 FOR i=1 TO 20
 90 PRINT AT i,0;"<CAPS 8>"
100 PRINT AT i,31;"<CAPS 8>"
110 NEXT i
120 FOR i=24 TO 170 STEP 8
130 PLOT 8,i: DRAW 239,0
140 NEXT i
150 FOR i=6 TO 250 STEP 8
160 PLOT i,24: DRAW 0,143
170 NEXT i
175 PAPER 1: INK 7
180 PRINT AT 2,3;"
                "
190 PRINT AT 3,3;"    DUELUL BR
OSCUTELOR      "
200 PRINT AT 4,3;"
                "
210 FOR i=1 TO 10
220 FOR j=a(i,1) TO a(i,2): PRI
NT AT 6+i,j;" ": NEXT j
230 PRINT AT 6+i,a(i,1);"<a>":
BEEP .02,-6: BEEP .04,-8
240 PRINT AT 6+i,a(i,2);"<b>":
BEEP .02,-6: BEEP .04,-8
250 NEXT i
255 FOR i=7 TO 16: PRINT AT i,4
; INK 1;"<CAPS 8>": NEXT i
260 LET sum=132
270 PRINT AT 20,1; PAPER 6; INK
 0;" Cine incepe (C=calc/J=juc)
? "
280 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
290 IF r$="c" THEN GO TO 550
300 IF r$="j" THEN GO TO 320
310 BEEP 1,-6: GO TO 280
320 PRINT AT 20,1; PAPER 6; INK
 0;"Astept mutarea ta (6 apoi 8-
9)": LET ind=0
325 PRINT AT 7,4; INK 7; PAPER
1;"<c>": LET s=7
330 PAUSE 0: LET r$=INKEY$
340 IF ind=1 THEN GO TO 360
350 IF r$="6" THEN GO TO 390
360 IF r$="8" THEN GO TO 430
370 IF r$="9" THEN GO TO 480
380 BEEP 1,-6: GO TO 330
390 BEEP .1,12: PRINT AT s,4; P
APER 1;" "
400 LET s=s+1: IF s=17 THEN LET
s=7
```

```
 410 PRINT AT s,4; INK 7; PAPER
1;"(c)"
 420 GO TO 330
 430 IF b(s-6)=0 THEN BEEP 1,-6:
 GO TO 330
 440 LET ind=1: PRINT AT s,a(s-6
,1); INK 1;"(CAPS 8)"
 450 PRINT AT s,a(s-6,1)+1;"(a)"
: BEEP .02,-6: BEEP .04,-8
 460 LET a(s-6,1)=a(s-6,1)+1: LE
T b(s-6)=b(s-6)-1: LET sum=sum-1
 470 GO TO 330
 480 BEEP .1,12: IF ind=0 THEN B
EEP 1,-6: GO TO 330
 485 PRINT AT s,4; PAPER 1;" ":
IF sum>0 THEN GO TO 550
 490 PRINT AT 20,1; PAPER 7; INK
 0;"Ai cistigat - felicitari !!!
"
 500 FOR i=1 TO 10: BEEP .1,i*4:
 NEXT i
 510 PRINT AT 20,1; PAPER 7; INK
 0;"  Alt joc (d/n) ?
"
 520 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 530 IF r$<>"d" THEN STOP
 540 PAPER 6: GO TO 10
 550 PRINT AT 20,1; INK 0; PAPER
 7;" Mutarea mea este urmatoarea
"
 560 LET bb=0
 570 FOR j=1 TO 5
 575 LET sb=0
 580 FOR i=1 TO 10
 590 LET nr=b(i)+1
 600 LET sb=sb+c(nr,j)
 610 IF sb=2 THEN LET sb=0
 620 NEXT i
 630 IF sb=0 THEN GO TO 730
 640 IF bb=1 THEN GO TO 700
 645 LET mut=0: LET bb=1
 650 FOR i=1 TO 10
 660 LET nr=b(i)+1
 670 IF c(nr,j)=0 THEN GO TO 690
 680 LET lin=i: LET mut=mut+m(j)
: GO TO 730
 690 NEXT i
 700 LET nr=b(lin)+1
 710 IF c(nr,j)=0 THEN LET mut=m
ut-m(j): GO TO 730
 720 LET mut=mut+m(j)
 730 NEXT j
 740 IF bb=1 THEN GO TO 810
 750 LET max=0
 760 FOR i=1 TO 10
 770 IF b(i)>max THEN LET max=b(
i): LET lin=i
 780 NEXT i
 790 IF max>=5 THEN LET mut=INT
(max/2): GO TO 810
 800 LET mut=1
 810 FOR i=1 TO mut
 820 PRINT AT lin+6,a(lin,2 -i+1
; INK 1;"(CAPS 8)"
 830 PRINT AT lin+6,a(lin,2)-i;"
(b)"
 840 BEEP .02,-6: BEEP .04,-8: N
EXT i
 850 LET b(lin)=b(lin)-mut
 860 LET a(lin,2)=a(lin,2)-mut
 870 LET sum=sum-mut
 880 IF sum>0 THEN GO TO 320
 890 PRINT AT 20,1; INK 0; PAPER
 7;"  Am cistigat !!!
"
 900 GO TO 500
 999 STOP
1000 DIM a(10,2): DIM b(10): DIM
 c(20,5): DIM m(5)
1010 FOR i=1 TO 10: READ a(i,1):
 READ a(i,2): NEXT i
1020 DATA 12,20,11,21,6,26,6,26,
7,25,8,24,11,21,12,20,8,24,8,24
1030 FOR i=1 TO 10: READ b(i): N
EXT i
1040 DATA 7,9,19,19,17,15,9,7,15
,15
1050 FOR i=1 TO 20: FOR j=1 TO 5
: READ c(i,j): NEXT j: NEXT i
1060 DATA 0,0,0,0,0,0,0,0,0,1,0,
0,0,1,0,0,0,0,1,1,0,0,1,0,0,0,0,
1,0,1,0,0,1,1,0,0,0,1,1,1,0,1,0,
0,0,0,1,0,0,1,0,1,0,1,0,0,1,0,1,
1,0,1,1,0,0,0,1,1,0,1,0,1,1,1,0,
0,1,1,1,1,0,0,0,0,1,0,0,0,1,1,
0,0,1,0,1,0,0,1,1
1070 FOR i=1 TO 5: READ m(i): NE
XT i
1080 DATA 16,8,4,2,1
1090 FOR i=0 TO 7: READ x: POKE
USR "a"+i,x: NEXT i
1100 DATA BIN 00000100,BIN 00001
010,BIN 01111110,BIN 11111100,BI
N 11011011,BIN 10101000,BIN 0111
0000,BIN 00010000
1110 FOR i=0 TO 7: READ x: POKE
USR "b"+i,x: NEXT i
1120 DATA BIN 00100000,BIN 01010
000,BIN 01111110,BIN 00111111,BI
N 11011011,BIN 00010101,BIN 0000
1110,BIN 00001000
1130 FOR i=0 TO 7: READ x: POKE
USR "c"+i,x: NEXT i
1140 DATA 0,16,24,28,31,28,24,16
1150 RETURN
```

TRAVERSARE

Jocul apare în setul FLEX produs de RECOOP și folosește o tablă de dimensiuni 8 × 8 și piese albe și negre; în FLEX, jocul este prezentat pentru șase piese de fiecare jucător, dar în program, pentru a scurta durata partidelor, se folosesc doar cîte cinci piese. Ele se așază la începutul partidei, ca în figura 2 (unde este precizat și modul de marcare al tablei, pentru indicarea mutărilor). Calculatorul mută piesele albe, în sus, iar jucătorul mută piesele negre, spre dreapta. O mutare constă în deplasarea

Fig.2



unei piese, cu o căsuță sau două, în direcția potrivită. Nu se poate sări peste o piesă a adversarului. Dacă o mutare ajunge într-un cîmp ocupat de o piesă a adversarului, aceasta este «bătută» și obligată să revină în cîmpul de pe care a plecat la începutul partidei.

Obiectivul jocului este scoaterea tuturor pieselor proprii prin partea opusă a tablei (calculatorul trebuie să traverseze, deci, tabla de jos în sus, iar jucătorul de la stînga la dreapta). Cel care reușește primul acest lucru cîștigă partida.

Programul întreabă «Cine joacă primul (C = calculatorul/ J = jucătorul)?», apoi se intră în partida propriu-zisă. Alegerea mutărilor calculatorului se face conform următoarelor priorități:

— poate fi bătută o piesă a adversarului?
— se poate scoate de sub atac o piesă proprie?
— se poate scoate o piesă de pe tablă?
— se poate muta fără a intra sub atacul unei piese a adversarului?

Se alege mutarea care corespunde primului răspuns afirmativ, parcurgînd aceste întrebări de sus în jos. Dacă răspunsul este negativ de fiecare dată, atunci se mută la întîmplare.

De reținut, deci, că nu se anticipează nici măcar o mutare a adversarului (ar mări mult timpul de răspuns), dar aceasta nu înseamnă că programul trebuie subestimat. Jucînd cu grijă, el poate fi, totuși, învins ușor.

După încheierea unei partide, jocul poate fi reluat (apăsînd tasta **D**).

Desigur, **modificarea** cea mai atractivă care i se poate aduce este aceea a anticipării mutărilor, eventual cu o opțiune inițială privind adîncimea acestei explorări a arborelui de evoluție a partidei.

Descrierea programului

20 — matricea **a** descrie tabla de joc.
30—60 — se completează cu 1 locul pieselor jucătorului (pe prima coloană a matricei **a**) și cu 2 locul pieselor calculatorului (pe linia de jos a matricei **a**).
80—160 — se desenează tabla.
170—200 — se desenează piesele, în poziția de start.

210 — **js** este numărul pieselor scoase de jucător, **cs** este numărul pieselor scoase de calculator.
220—225 — opţiunea primei mutări.
230—270 — se înregistrează şi se verifică mutarea jucătorului: linia şi numărul de paşi cu care se deplasează piesa de pe linia respectivă.
290—300 — se caută piesa de pe linia indicată (coloana **j**).
305 — nu există nici o piesă.
310 — piesa iese de pe linie.
313 — deplasare peste o piesă a adversarului — eroare.
315—330 — efectuarea mutării.
360—390 — se bate o piesă a calculatorului.
410 — la scoaterea unei piese, se verifică dacă mai sînt piese în joc.
415—445 — mesaj de cîştig pentru jucător, opţiune de reluare.
452—480 — se caută o posibilitate de a bate o piesă a adversarului, din partea dreaptă a tablei (pe cale de a fi scoasă).
490—530 — se caută o piesă proprie atacată.
545—550 — precizarea parametrilor mutării în cazul anterior.
560—580 — se încearcă scoaterea unei piese de pe tablă.
590—635 — se caută o posibilitate de a bate o piesă a adversarului, pe mijlocul tablei.
640—685 — se caută o mutare care să nu cadă sub atacul unei piese a jucătorului, plecînd de pe linia 3 în sus (se pregăteşte, deci, scoaterea unei piese de pe tablă).
690—730 — se face o mutare oarecare, la mijlocul tablei.
740—790 —se caută o mutare care să nu cadă sub atacul unei piese a adversarului, în partea de jos a tablei.
800—840 — se face o mutare oarecare, în partea din dreapta a tablei, începînd de jos în sus.
2000—3090 — se efectuează mutarea calculatorului; dacă piesa iese de pe tablă (**i**≤**p**), atunci se măreşte variabila **cs** şi se compară cu 5.
2050 — a cîştigat calculatorul.
3000 —3060 — mutare pe tablă.
3070—3080 — se bate o piesă a adversarului.

```
  10 BORDER 1: PAPER 6: CLS : IN
K 0
  20 DIM a(8,8)
  30 FOR i=1 TO 5
  40 LET a(i,1)=1
  50 LET a(8,3+i)=2
  60 NEXT i
  70 FOR i=0 TO 8
  80 PLOT 76,36+i*16: DRAW 128,0
  90 PLOT 76+i*16,36: DRAW 0,128
 100 NEXT i
 110 PLOT 77,168: DRAW 15,0: DRA
W -3,-3: PLOT 92,168: DRAW -3,3
 120 PLOT 209,37: DRAW 0,15: DRA
W -3,-3: PLOT 209,52: DRAW 3,-3
 130 FOR i=1 TO 8
 140 PRINT AT 18,8+i*2;i
 150 PRINT AT i*2,8;i
 160 NEXT i
 170 FOR i=4 TO 8
 180 PRINT AT 16,8+2*i;"O"
 190 PRINT AT 2*i-6,10; INVERSE
1;"O"
 200 NEXT i
 210 LET js=0: LET cs=0
 220 PRINT AT 21,1;"CINE INCEPE(
J=juc/C=calc) ?": PAUSE 0: LET r
$=INKEY$: BEEP .1,12
```

```
 225 IF r$="c" THEN GO TO 450
 230 PRINT AT 21,1;"MUTAREA TA:
LINIA=          "
 240 PAUSE 0: LET i$=INKEY$: BEE
P .1,12: BEEP .2,22
 245 IF i$>="1" AND i$<="5" THEN
 LET i=VAL i$: GO TO 250
 247 PRINT AT 21,1;"MUTARE ERONA
TA             ": BEEP .2,0: BE
EP .1,-6: PAUSE 50: GO TO 230
 250 PRINT AT 21,20;i;" PASI="
 260 PAUSE 0: LET p$=INKEY$: BEE
P .1,12: BEEP .2,22
 265 IF p$<"1" OR p$>"2" THEN GO
 TO 247
 270 LET p=VAL p$: PRINT AT 21,2
8;p
 290 FOR j=1 TO 8
 295 IF a(i,j)=1 THEN GO TO 310
 300 NEXT j
 305 GO TO 247
 310 IF j=8 THEN GO TO 315
 313 IF p=2 AND a(i,j+1)=2 THEN
GO TO 247
 315 LET a(i,j)=0
 320 PRINT AT 2*i,8+2*j; FLASH 1
;"O": PAUSE 80: PRINT AT 2*i,8+2
*j;" "
```

```
325 IF j+p>8 THEN GO TO 410
330 PRINT AT 2*i,8+2*(j+p); FLASH 1;"O": PAUSE 80: PRINT AT 2*i,8+2*(j+p); INVERSE 1;"O"
360 IF a(i,j+p)=2 THEN GO TO 380
370 LET a(i,j+p)=1: GO TO 450
380 LET a(i,j+p)=1: LET a(8,j+p)=2
390 PRINT AT 16,8+2*(j+p);"O": GO TO 450
410 BEEP .05,13: BEEP .1,22: BEEP .4,13: LET js=js+1: IF js<5 THEN GO TO 450
415 PRINT AT 21,1;"FELICITARI - AI CISTIGAT      "
420 FOR i=1 TO 30
425 BEEP .03,INT (RND*20)
430 NEXT i: PAUSE 80
435 PRINT AT 21,1;"Alt joc (d/n) ?
440 PAUSE 0: IF INKEY$<>"d" THEN STOP
445 CLS : GO TO 10
450 PRINT AT 21,1;"MUTAREA MEA "; FLASH 1;"<6>"; FLASH 0;"         "
452 FOR r=1 TO 5
455 FOR k=4 TO 8
460 IF a(r,k)=1 AND a(1+r,k)=2 THEN LET i=1+r: LET j=k: LET p=1: GO TO 2000
465 IF a(r,k)=1 AND a(2+r,k)=2 THEN LET i=2+r: LET j=k: LET p=2: GO TO 2000
470 NEXT k
480 NEXT r
490 FOR k=3 TO 5
500 FOR l=4 TO 8
510 IF a(k,l)=2 AND (a(k,l-1)=1 OR a(k,l-2)=1) THEN GO TO 545
520 NEXT l
530 NEXT k
540 GO TO 560
545 IF a(k-2,l-1)<>1 AND a(k-2,l-2)<>1 THEN LET i=k: LET j=l: LET p=2: GO TO 2000
550 IF a(k-1,l-1)<>1 AND a(k-1,l-2)<>1 THEN LET i=k: LET j=l: LET p=1: GO TO 2000
560 FOR k=4 TO 8
570 IF a(1,k)=2 THEN LET i=1: LET j=k: LET p=1: GO TO 2000
575 IF a(2,k)=2 THEN LET i=2: LET j=k: LET p=2: GO TO 2000
580 NEXT k
590 FOR r=0 TO 3
600 FOR k=3 TO 5
610 IF a(k,6-r)=1 AND a(k+1,6-r)=2 THEN LET i=k+1: LET j=6-r: LET p=1: GO TO 2000
620 IF a(k,6-r)=1 AND a(k+2,6-r)=2 THEN LET i=k+2: LET j=6-r: LET p=2: GO TO 2000
630 NEXT k
635 NEXT r
640 FOR k=4 TO 8
660 IF a(3,k)=2 AND a(1,k-1)<>1 AND a(1,k-2)<>1 THEN LET i=3: LET j=k: LET p=2: GO TO 2000
670 IF a(3,k)=2 AND a(2,k-1)<>1 AND a(2,k-2)<>1 THEN LET i=3: LET j=k: LET p=1: GO TO 2000
680 IF a(4,k)=2 AND a(2,k-1)<>1 AND a(2,k-2)<>1 THEN LET i=4: LET j=k: LET p=2: GO TO 2000
685 NEXT k
690 FOR k=3 TO 6
700 FOR l=1 TO 8
710 IF a(k,l)=1 THEN GO TO 730
720 IF a(k+2,l)=2 THEN LET i=k+2: LET j=l: LET p=2: GO TO 2000
725 IF a(k+1,l)=2 THEN LET i=k+1: LET j=l: LET p=1: GO TO 2000
727 NEXT l
730 NEXT k
740 FOR k=4 TO 8
750 FOR l=8 TO 5 STEP -1
760 IF a(k,l)=2 AND a(k-2,l-1)<>1 AND a(k-2,l-2)<>1 THEN LET i=k: LET j=l: LET p=2: GO TO 2000
770 IF a(k,l)=2 AND a(k-1,l-1)<>1 AND a(k-1,l-2)<>1 THEN LET i=k: LET j=l: LET p=1: GO TO 2000
780 NEXT l
790 NEXT k
800 FOR k=8 TO 1 STEP -1
810 FOR l=4 TO 8
820 IF a(k,l)=2 THEN LET i=k: LET j=l: LET p=2: GO TO 2000
830 NEXT l
840 NEXT k
2000 BEEP .1,12: BEEP .2,22
2010 PRINT AT 2*i,8+2*j; FLASH 1;"O"
2020 PAUSE 80: PRINT AT 2*i,8+2*j;" "
2025 LET a(i,j)=0
2030 IF i>p THEN GO TO 3000
2035 BEEP .05,13: BEEP .01,22: BEEP .5,13
2040 LET cs=cs+1: IF cs<5 THEN GO TO 230
2050 PRINT AT 21,1;"AM CISTIGAT        ": GO TO 420
3000 LET i=i-p
3050 PRINT AT 2*i,8+2*j; FLASH 1;"O": PAUSE 80
3060 PRINT AT 2*i,8+2*j;"O"
3070 IF a(i,j)=0 THEN GO TO 3090
3080 LET a(i,1)=1: PRINT AT 2*i,10; INVERSE 1;"O"
3090 LET a(i,j)=2: GO TO 230
```

IMPAS

Jocul este prezentat în setul FLEX, realizat de RECOOP, pentru tabla de de dimensiuni 8 × 8, cu 16 piese, dar programul vă propune să jucați pe tabla 6 × 6, cu 9 piese. Poziția de plecare este cea din figura 3.

Jucătorul mută în sus, calculatorul spre dreapta. O mutare constă în deplasarea unei piese (toate au aceeași culoare) cu oricîți pași dorim, în direcția potrivită, fără a sări însă peste o altă piesă. Jucătorul care, aflat la rînd, nu mai poate muta, pierde partida.

Pentru indicarea unei mutări, programul cere coloana (literă între **a** și **f**), linia (cifră între **1** și **6**) și numărul de pași pe care doriți să-i facă piesa cu coordonatele specificate.

În alegerea mutărilor sale, programul caută cea mai scurtă deplasare de piesă care lasă adversarul cu numărul cel mai mic de mutări la pasul imediat următor. Criteriul nu este, desigur, suficient pentru a-i asigura o tărie deosebită. Mai exact, programul poate juca bine prima parte a partidei, dar în final poate fi « păcălit » ușor, el neputînd anticipa mutările adversarului. De exemplu, dacă el poate învinge într-o mutare, găsește acea mutare, dar nu poate găsi calea de a cîștiga în două sau mai multe mutări. În plus, încercînd să diminueze posibilitățile de mișcare ale adversarului, nu acordă suficientă atenție posibilităților proprii de mișcare.

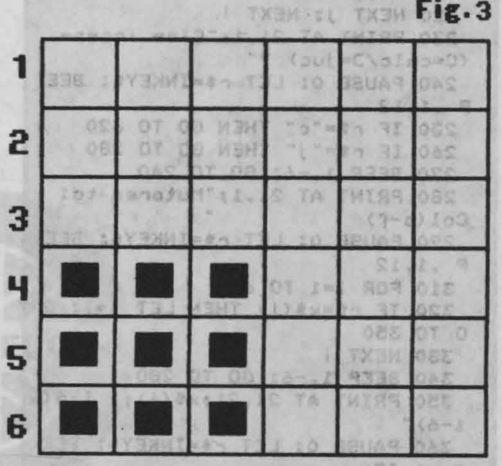
O **modificare** care să-i dea un plus de tărie poate consta în anticiparea mai multor mutări (cu alegerea acelei mutări care diminuează la distanță posibilitățile adversarului). Evident, timpul de răspuns va crește considerabil.

La începutul fiecărei partide, programul întreabă « Cine mută primul (C = calculatorul/J = jucătorul)? », iar după partidă întreabă dacă se dorește « Alt joc (d/n)? »

Descrierea programului

40—70 — se desenează tabla de joc.
90—120 — se marchează tabla.
190 — matricea **a** va descrie tabla la un moment dat al partidei (cîmpurile ocupate sînt indicate prin cifra 1, cele libere prin cifra zero).
200 — 220 — completarea matricei **a**.
230—270 — opțiune de începere a jocului (cine face prima mutare?).
280—470 — se introduce mutarea jucătorului: piesa de mutat (i se dau coordonatele **i** și **j**) și numărul de pași (variabila **p**); în același timp, se verifică dacă mutarea este posibilă (linia 390, liniile 440—470).
480—510 — efectuarea mutării.

Fig. 3



515 — înregistrarea mutării.
520—535 — se verifică dacă mai există posibilități de mutare pentru calculator.
540—610 — mesaj de victorie pentru jucător și opțiune de reluare.
630—850 — se caută mutarea optimă (de coordonate **io, jo** și cu **po** pași) în sensul diminuării maxime a numărului de mutări care îi mai rămîn jucătorului; pentru fiecare mutare posibilă, pe rînd (liniile 650—690), se evaluează numărul de deplasări (variabila **dep**) posibile la momentul următor pentru jucător (liniile 700—780); dacă se găsește o mutare cîștigătoare, se alege aceea (linia 820).

860—900 — se efectuează mutarea calculatorului.

905 — se înregistrează mutarea calculatorului.

910—930 — se verifică dacă jucătorul mai are posibilități de mutare; în caz afirmativ, se merge la linia 280 (o nouă mutare).

940 — victoria calculatorului.

```
  10 BORDER 1: PAPER 6: CLS : PR
INT AT 10,10; FLASH 1;"I M P A S
"
  20 FOR i=1 TO 20: BEEP .03,INT
 (RND*30): NEXT i
  30 PAUSE 50: CLS
  40 FOR i=0 TO 6
  50 PLOT 56,24+i*24: DRAW 144,0
  60 PLOT 56+24*i,24: DRAW 0,144
  70 NEXT i
  80 DIM x$(6): LET x$="abcdef"
  90 FOR i=1 TO 6
 100 PRINT AT -1+3*i,6;i
 110 PRINT AT 19,5+3*i;x$(i)
 120 NEXT i
 130 PLOT 214,90: DRAW 0,-60
 140 PLOT 220,86: DRAW -6,6: DRA
W -6,-6
 150 DIM y$(7): LET y$="JUCATOR"
 160 FOR i=1 TO 7
 170 PRINT AT 10+i,28;y$(i)
 180 NEXT i
 190 DIM a(6,6)
 200 FOR i=4 TO 6: FOR j=1 TO 3
 210 LET a(i,j)=1: PRINT AT -1+3
*i,5+3*j;"<CAPS 8>"
 220 NEXT j: NEXT i
 230 PRINT AT 21,1;"Cine incepe
(C=calc/J=juc) ?"
 240 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 250 IF r$="c" THEN GO TO 620
 260 IF r$="j" THEN GO TO 280
 270 BEEP 1,-6: GO TO 240
 280 PRINT AT 21,1;"Mutarea ta:
Col(a-f)                "
 290 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 310 FOR i=1 TO 6
 320 IF r$=x$(i) THEN LET j=i: G
O TO 350
 330 NEXT i
 340 BEEP 1,-6: GO TO 280
 350 PRINT AT 21,21;x$(i);" lin(
1-6)"
 360 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 370 IF r$>="1" AND r$<="6" THEN
 LET i=VAL r$: GO TO 390
 380 BEEP 1,-6: GO TO 350
 390 IF a(i,j)=0 THEN GO TO 340
 395 PRINT AT 21,1;"Citi pasi (1
 - 6) ?                  "
 400 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 410 IF r$>="1" AND r$<="6" THEN
 LET p=VAL r$: GO TO 440
 420 BEEP 1,-6: GO TO 280
 440 IF p>=i THEN GO TO 420
 450 FOR k=i-p TO i-1
 460 IF a(k,j)=1 THEN GO TO 340
 470 NEXT k
 480 PRINT AT -1+3*i,5+3*j; FLAS
H 1;"<6>"
 490 PRINT AT -1+3*(i-p),5+3*j;
FLASH 1;"<6>"
 500 PAUSE 60: PRINT AT -1+3*i,5
+3*j;" "
 510 PRINT AT -1+3*(i-p),5+3*j;"
<CAPS 8>"
 515 LET a(i,j)=0: LET a(i-p,j)=
1
 520 FOR i=1 TO 6: FOR j=1 TO 5
 530 IF a(i,j)=1 AND a(i,j+1)=0
THEN GO TO 620
 535 NEXT j: NEXT i
 540 PRINT AT 21,1;"Ai cistigat
- felicitari !!        "
 550 FOR i=1 TO 5: FOR j=7 TO 1
STEP -1
 560 BORDER j: BEEP .03,i*j+10
 570 NEXT j: NEXT i
 580 PAUSE 40: PRINT AT 21,1;"Al
t joc (d/n) ?                   "
 590 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 600 IF r$<>"d" THEN STOP
 610 GO TO 10
 620 PRINT AT 21,1;"Mutarea mea
                "
 630 LET io=1: LET jo=1
 640 LET po=0: LET depo=1000
 650 FOR i=1 TO 6: FOR j=1 TO 5
 660 IF a(i,j)=0 OR a(i,j+1)=1 T
```

```
HEN GO TO 840
 670 FOR k=j+1 TO 6
 680 IF a(i,k)=1 THEN GO TO 840
 690 LET a(i,j)=0: LET a(i,k)=1
 700 LET dep=0
 710 FOR t=2 TO 6: FOR s=1 TO 6
 720 IF a(t,s)=0 THEN GO TO 770
 730 FOR v=t-1 TO 1 STEP -1
 740 IF a(v,s)=1 THEN GO TO 770
 750 LET dep=dep+1
 760 NEXT v
 770 NEXT s
 780 NEXT t
 790 LET a(i,j)=1: LET a(i,k)=0
 800 IF depo<=dep THEN GO TO 840
 810 LET depo=dep: LET io=i: LET
 jo=j: LET po=k-j
 820 IF depo=0 THEN GO TO 860
 830 NEXT k
 840 NEXT j
```

```
 850 NEXT i
 860 BEEP .1,12: BEEP .05,22: LE
T a(io,jo)=0: LET a(io,jo+po)=1
 870 PRINT AT -1+3*io,5+3*jo; FL
ASH 1;"(6)"
 880 PRINT AT -1+3*io,5+3*(jo+po
); FLASH 1;"(6)"
 890 PAUSE 60: PRINT AT -1+3*io,
5+3*jo;" "
 900 PRINT AT -1+3*io,5+3*(jo+po
);"(CAPS 8)"
 905 LET a(io,jo)=0: LET a(io,jo
+po)=1
 910 FOR i=2 TO 6: FOR j=1 TO 6
 920 IF a(i,j)=1 AND a(i-1,j)=0
THEN GO TO 280
 930 NEXT j: NEXT i
 940 PRINT AT 21,1;"Am cistigat
!!!              ": GO TO 550
```

jocul evitării pătratelor

Jocul este inventat de Martin Gardner (a se vedea cartea sa **Alte amuzamente matematice,** Editura Științifică, București, 1970) și, după cum spune și numele, se bazează pe principiul « cine realizează o anume formație de piese pierde partida ». În cazul nostru, formația de evitat este pătratul, patru piese plasate în colțurile unui pătrat de orice dimensiune, așezat în orice poziție și în orice orientare pe tablă. În program se folosește o tablă de joc de dimensiuni 5×5. Figura 4 alăturată indică cîteva pătrate — dintre cele 50 — posibile pe această tablă.

În program, coloanele tablei sînt marcate cu literele **a, b, c, d, e,** iar liniile cu cifrele **1, 2, 3, 4, 5;** cîmpul din stînga-jos este **a1** (ca la șah). Indicarea unei mutări se face precizînd coloana (o literă) și linia (o cifră) cîmpului în care se dorește plasarea unei piese. Înainte de joc, programul întreabă « Cine mută primul (C = calculatorul / J = jucătorul) ? », iar în final întreabă « Alt joc (d/n)? ».

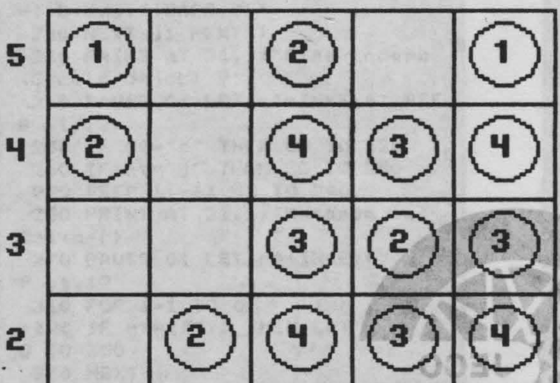


Fig.4

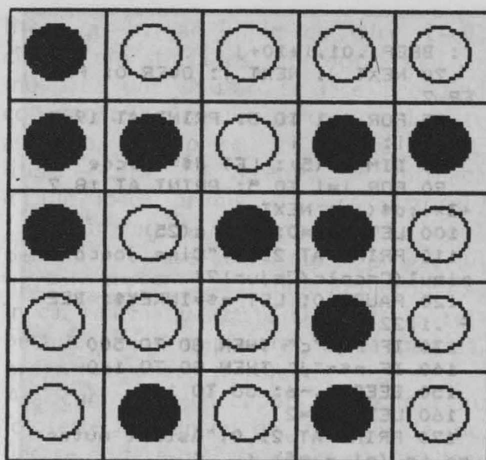
Programul joacă mai degrabă « la greșeala adversarului » decît conform unui plan propriu. Evident, el nu mută niciodată pentru a forma un pătrat (cînd nu mai are nici o mutare prin care nu pierde imediat, se recunoaște învins) și verifică de fiecare dată dacă adversarul a pierdut. Singurul principiu strategic pe care îl urmărește este acela de a nu muta într-un cîmp care completează pentru adversar un pătrat (cînd există, deci, piese ale adversarului în celelalte trei colțuri). Bineînțeles, dacă nu poate face altfel, joacă și în asemenea puncte.

Jucînd atent, programul poate fi învins. Jocul se poate termina și remiză, prin umplerea tablei, fără ca un jucător să fi realizat un pătrat — a se vedea figura 5.

Probabil că un plus de tărie se poate adăuga programului (fără a privi totuși înainte, pe arborele de evoluție a jocului) suplimentîndu-i strategia cu încă un principiu: la fiecare mutare să evite crearea de « auto-capcane », tri-

plete de piese în colțurile unui pătrat (astfel încît ocuparea celui de-al patrulea colț să conducă la pierderea partidei). Sarcina de a face această îmbunătățire rămîne în seama cititorului.

Fig. 5



Descrierea programului

5.— variabila **z** este destinată evitării reparcurgerii subrutinei 1000 la reluarea programului (economie de timp).
30—50 — se desenează tabla.
60—70 — se colorează ecranul.
75—90 — se marchează tabla.
100 — **mut** numără mutările, iar **a** conține tabla de joc sub formă de vector.
160 — **juc** specifică jucătorul aflat la mutare (1 = calculator, 2 = jucător).
170—320 — se așteaptă mutarea jucătorului, se verifică și se execută; vectorul **a$** conține piesele: albă pentru calculator, pe prima poziție (« a » în modul grafic, desenat la liniile 1100—1110) și neagră pentru jucător, pe poziția a doua (« b » în modul grafic, desenat la liniile 1120—1130); completarea vectorului **a$** se face la linia 1140.
320 (GOSUB 1200) — subrutina 1200 verifică dacă s-a completat un pătrat de către jucătorul **juc**; pătratele sînt memorate în matricea **p**, la liniile 1010—1040; **ind** = 1 indică realizarea unui pătrat, deci încheierea partidei.
340—410 — se marchează pătratul realizat de jucător.
420—450 — mesajul de cîștig și opțiunea de reluare.
505 — pentru alegerea mutării sale, calculatorul face două explorări ale tablei (ale vectorului **a**): la prima (**tr** = **1**) încearcă să mute în așa fel încît să nu completeze un pătrat, cu trei colțuri ocupate de adversar; dacă nu se poate (**tr** = 2), atunci mută și în asemenea puncte.
520—580 — în cursul primelor 10 mutări ale partidei, programul alege locul de jucat la întîmplare, conform însă mențiunilor dinainte (la prima trecere nu completează pătrate adverse — liniile 544—548), fără a pierde însă (liniile 550—570).
590—640 — după mutarea a 10-a a partidei, se caută sistematic (începînd din stînga-sus) un loc pentru mutare, din nou evitînd formarea unui patrat (liniile 610—630) și completarea unui pătrat, cu trei colțuri ocupate de adversar (liniile 604—608).
645—660 — programul își anunță înfrîngerea.
670—710 — se execută o mutare a calculatorului (**lin, col** sînt coordonatele pe ecran ale piesei, în funcție de valoarea variabilei **loc,** care indică locul mutării în vectorul **a**).

```
  5 LET z=0
 10 CLS : BORDER 1
 20 PRINT AT 1,1; FLASH 1;"  JO
CUL EVITARII PATRATELOR   "
 25 GO SUB 1000
 30 FOR i=1 TO 6
```

```
 40 PLOT 72,8+24*i: DRAW 120,0
 45 PLOT 48+24*i,32: DRAW 0,120
 50 NEXT i: OVER 1: PAPER 2
 60 FOR i=0 TO 21: FOR j=0 TO 2
8 STEP 4
 65 PRINT AT i,j; PAPER 4;"
```

```
": BEEP .01,i+10+j
  70 NEXT j: NEXT i: OVER 0: PAP
ER 7
  75 FOR i=1 TO 5: PRINT AT 19-3
*i,8;i: NEXT i
  80 DIM d$(5): LET d$="abcde"
  90 FOR i=1 TO 5: PRINT AT 18,7
+3*i;d$(i): NEXT i
 100 LET mut=0: DIM a(25)
 110 PRINT AT 21,0;"Cine joaca p
rimul(C=calc/J=juc)?"
 120 PAUSE 0: LET r$=INKEY$: BEE
P .1,22
 130 IF r$="c" THEN GO TO 500
 140 IF r$="j" THEN GO TO 160
 150 BEEP 1,-6: GO TO 110
 160 LET juc=2
 170 PRINT AT 21,0;"Astept mutar
ea ta (a1 - e5) :      "
 180 PAUSE 0: LET r$=INKEY$: BEE
P .1,22: PRINT AT 21,29;r$
 190 PAUSE 0: LET t$=INKEY$: BEE
P .1,22: PRINT AT 21,30;t$
 200 FOR i=1 TO 5
 210 IF d$(i)=r$ THEN LET col=i:
 GO TO 240
 220 NEXT i
 230 BEEP 1,-6: GO TO 170
 240 IF t$<"1" OR t$>"5" THEN GO
 TO 230
 250 LET lin=VAL t$
 270 LET loc=(lin-1)*5+col
 280 IF a(loc)<>0 THEN GO TO 230
 290 LET a(loc)=2: LET mut=mut+1
 300 PRINT AT 19-3*lin,7+3*col;
FLASH 1;a$(2)
 310 PAUSE 90: BEEP .3,22: PRINT
 AT 19-3*lin,7+3*col;a$(2)
 320 PRINT AT 21,0;"
                  ": GO SUB 12
00
 330 IF ind=0 THEN GO TO 460
 340 FOR k=1 TO 4
 350 LET lin=INT ((p(u,k)-1)/5)+
1
 360 LET col=p(u,k)-5*(lin-1)
 370 PRINT AT 19-3*lin,7+3*col;
FLASH 1;a$(2)
 380 NEXT k
 390 FOR i=1 TO 5: FOR j=1 TO 7
 400 BORDER j: BEEP .02,i*j
 410 NEXT j: NEXT i
 420 PRINT AT 19,2;"AM CISTIGAT
 !!!"
 425 PRINT AT 21,0;"Alt joc (d/n
) ?                   "
 430 PAUSE 0: LET r$=INKEY$: BEE
P .1,22
 440 IF r$<>"d" THEN STOP
 450 BEEP .3,22: GO TO 10
 460 IF mut<25 THEN GO TO 500
 470 PRINT AT 19,2;"REMIZA"
 480 BEEP .2,12: BEEP .4,22: BEE
P 1.33
 490 GO TO 425
 500 PRINT AT 21,0;"E rindul meu
 - asteapta putin ! ": LET juc=1
 505 LET tr=1
 510 IF mut>10 THEN GO TO 590
 520 FOR i=1 TO 6
 530 LET loc=INT (RND*25)+1
 540 IF a(loc)<>0 THEN GO TO 580
 542 IF tr=2 THEN GO TO 550
 544 LET a(loc)=2: LET juc=2: GO
 SUB 1200
 546 LET juc=1: LET a(loc)=0
 548 IF ind=1 THEN GO TO 580
 550 LET a(loc)=1: GO SUB 1200
 560 IF ind=0 THEN GO TO 670
 570 LET a(loc)=0
 580 NEXT i
 590 FOR i=1 TO 25
 600 IF a(i)<>0 THEN GO TO 640
 602 IF tr=2 THEN GO TO 610
 604 LET loc=i: LET a(loc)=2: LE
T juc=2: GO SUB 1200
 606 LET juc=1: LET a(loc)=0
 608 IF ind=1 THEN GO TO 640
 610 LET loc=i: LET a(loc)=1: GO
 SUB 1200
 620 IF ind=0 THEN GO TO 670
 630 LET a(loc)=0
 640 NEXT i
 645 IF tr=1 THEN LET tr=2: GO T
O 590
 650 BEEP 1,-6: BEEP .5,8: BEEP
.8,-8
 660 PRINT AT 19,2;"AI CISTIGAT
- FELICITARI !!": GO TO 425
 670 LET mut=mut+1
 680 LET lin=INT ((loc-1)/5)+1
 690 LET col=loc-(lin-1)*5
 700 PRINT AT 19-3*lin,7+3*col;
FLASH 1;a$(1)
 710 PAUSE 90: BEEP .3,22: PRINT
 AT 19-3*lin,7+3*col;a$(1)
 720 IF mut<25 THEN GO TO 160
 730 GO TO 470
 999 STOP
1000 IF z<>0 THEN RETURN
1010 DIM p(50,4): LET z=1
1020 FOR i=1 TO 50: FOR j=1 TO 4
: READ p(i,j): NEXT j: NEXT i
1025 DATA 1,2,6,7,2,3,7,8,3,4,8,
9,4,5,9,10,6,7,11,12,7,8,12,13,8
,9,13,14,9,10,14,15,11,12,16,17,
12,13,17,18,13,14,18,19,14,15,19
,20,16,17,21,22,17,18,22,23,18,1
9,23,24,19,20,24,25
1030 DATA 1,3,11,13,2,4,12,14,3,
5,13,15,6,8,16,18,7,9,17,19,8,10
,18,20,11,13,21,23,12,14,22,24,1
3,15,23,25,1,4,16,19,2,5,17,20,6
,9,21,24,7,10,22,25,1,5,21,25
1040 DATA 6,2,12,8,7,3,13,9,8,4,
14,10,11,7,17,8,12,8,18,14,13,9,
19,15,16,12,22,18,17,13,23,19,18
,14,24,20,11,3,23,15,6,3,14,17,7
```

```
,4,15,18,11,8,19,22,12,9,20,23,1
1,2,9,18,12,3,10,19,16,7,14,23,1
7,8,15,24,6,4,20,22,2,10,24,16
1100 FOR i=0 TO 7: READ x: POKE
USR "a"+i,x: NEXT i
1110 DATA 60,66,129,129,129,129,
66,60
1120 FOR i=0 TO 7: READ x: POKE
USR "b"+i,x: NEXT i
1130 DATA 60,126,255,255,255,255
,126,60
1140 DIM a$(2): LET a$="<ab>":
1150 RETURN
1200 LET ind=0
1210 FOR u=1 TO 50: FOR v=1 TO 4
1220 IF loc<>p(u,v) THEN GO TO 1
270
1230 FOR k=1 TO 4
1240 IF a(p(u,k))<>juc THEN GO T
O 1270
1250 NEXT k
1260 LET ind=1: RETURN
1270 NEXT v
1280 NEXT u: RETURN
```

JOCUL DISTANȚELOR

Pe o tablă de dimensiuni 8×8, doi jucători așază pe rînd cîte o piesă (amîndoi au piese de aceeași culoare), în cîmpuri libere. Pierde partida cel care realizează două perechi de piese echidistante, orizontal sau vertical (o pereche poate fi orizontală și cealaltă verticală și, de asemenea, o piesă poate apărea în ambele perechi).

Tabla este marcată ca la șah (coloanele cu litere **a—h**, iar liniile cu cifre **1—8**; cîmpul din stînga-jos este notat **a1**), iar indicarea unei mutări se face precizînd coordonatele cîmpului pe care se dorește așezarea unei piese, în această formă, literă — cifră. La început, se întreabă « Cine joacă primul (C = calculatorul / J = jucătorul)? », iar la realizarea a două perechi de piese echidistante, programul întreabă « Vrei să vezi unde ai pierdut? »; în caz afirmativ (tasta **D**), piesele respective devin clipitoare. După partidă, programul poate fi reluat.

Așa cum este conceput, programul nu folosește nici un fel de strategie de joc, fiecare mutare pe care o face evitînd doar pierderea partidei (dacă se poate). Cu toate acestea, el nu este un partener banal, deoarece . . . nu greșește niciodată, ceea ce jucătorului i se poate ușor întîmpla. Desigur, o **modificare** care ar putea fi atractivă este aceea a considerării de piese de culori diferite pentru cei doi jucători și pierderea partidei de către cel care realizează cu piesele proprii două perechi echidistante. O altă modificare de interes poate fi considerarea unei variante a jocului în care un jucător pierde atunci cînd realizează **trei** perechi de piese proprii echidistante, ceea ce mărește durata unei partide și dificultatea alegerii mutărilor.

Descrierea programului

20 — tabla de joc este descrisă de matricea **a**, iar vectorul **d** conține distanțele deja realizate.

30—140 — desenarea și marcarea tablei.

150—200 — alegerea celui care mută primul.

210 — **juc** = 1 indică faptul că mută jucătorul, **juc** = 2 indică faptul că la mutare este calculatorul.

210—330 — se cere jucătorului să facă o mutare, se analizează dacă este corectă și dacă locul nu este ocupat, apoi se efectuează mutarea (în cîmpul de coordonate **lin, col**).

340 — 390 — se testează dacă nu s-a format o nouă pereche pe linie.

400—450 — se testează dacă nu s-a format

o pereche nouă pe coloană; în caz afirmativ, se mărește de fiecare dată componenta corespunzătoare a vectorului **d** (liniile 380, respectiv, 440).
460—480 — se verifică dacă s-au realizat două perechi echidistante.
500—540 — mesaj anunțînd victoria calculatorului.
550—850 — dacă jucătorul dorește (este întrebat acest lucru la liniile 550—620), i se arată unde a pierdut; pentru aceasta, se caută perechi orizontale (liniile 660—740) sau verticale (liniile 760—840) de piese aflate la distanța **dist,** care apare de două ori (**d(dist)** ⩾ 2 la liniile 470 și 640).
910 — vectorul **f** indică distanțele care ar fi formate la mutarea următoare a calculatorului.
920—1080 — se încearcă de 5 ori efectuarea unor mutări în poziții alese la întîmplare (linia 920), dacă nu se realizează perechi echidistante; pentru aceasta, se caută perechi orizontale (liniile 940—1000) sau verticale (liniile 1010—1060) și se acceptă mutarea în cîmpul de coordonate **lin, col** numai dacă nu se pierde partida (liniile 1062—1070).
1081—1250 — se caută sistematic, începînd din stînga-sus, o mutare posibilă, căutînd din nou perechi orizontale (liniile 1110—1160) și verticale (liniile 1170—1220); mutarea este acceptată numai dacă nu se pierde partida (liniile 1222—1230).
1255—1300 — nici o mutare nu este posibilă fără a pierde; se comunică acest lucru.
1310—1340 — efectuarea mutării calculatorului, în poziția (**lin, col**).

```
  10 PAPER 6: BORDER 1: CLS
  20 DIM a(8,8): DIM d(7)
  30 DIM b$(8): LET b$="abcdefgh
"
  70 FOR i=1 TO 9
  80 PLOT 44+16*i,36: DRAW 0,128
  90 PLOT 60,20+16*i: DRAW 128,0
 100 NEXT i
 110 FOR i=1 TO 8
 120 PRINT AT 2*i,6;9-i
 130 PRINT AT 18,6+2*i;b$(i)
 140 NEXT i
 150 PRINT AT 21,1;"Cine incepe
(J=juc/C=calc) ?"
 160 PAUSE 0: LET c$=INKEY$: BEE
P .1,12: BEEP .1,22
 170 IF c$="j" THEN GO TO 210
 180 IF c$="c" THEN GO TO 900
 190 BEEP 1,-6
 200 GO TO 150
 210 LET juc=1: PRINT AT 21,1;"M
utarea ta (a1 - h8) "; FLASH 1;"
?"; FLASH 0;"     "
 220 PAUSE 0: LET c$=INKEY$: BEE
P .1,12: BEEP .1,22
 230 IF c$<"a" OR c$>"h" THEN BE
EP 1,-6: GO TO 220
 240 PRINT AT 21,22;c$
 250 PAUSE 0: LET l$=INKEY$: BEE
P .1,12: BEEP .1,22
 260 IF l$<"1" OR l$>"8" THEN BE
EP 1,-6: GO TO 250
 265 PRINT AT 21,23;l$
 270 LET lin=9-VAL l$
 280 FOR i=1 TO 8
 290 IF c$=b$(i) THEN LET col=i:
GO TO 310
```

```
 300 NEXT i
 310 IF a(lin,col)=1 THEN BEEP 1
,-6: GO TO 210
 315 LET a(lin,col)=1
 320 PRINT AT 2*lin,6+2*col; FLA
SH 1;"<6>"
 330 PAUSE 80: PRINT AT 2*lin,6+
2*col;"<CAPS 8>"
 340 FOR i=1 TO 8
 350 IF i=col THEN GO TO 390
 360 IF a(lin,i)=0 THEN GO TO 39
0
 370 LET dist=ABS (i-col)
 380 LET d(dist)=d(dist)+1
 390 NEXT i
 400 FOR i=1 TO 8
 410 IF i=lin THEN GO TO 450
 420 IF a(i,col)=0 THEN GO TO 45
0
 430 LET dist=ABS (i-lin)
 440 LET d(dist)=d(dist)+1
 450 NEXT i
 455 IF juc=2 THEN GO TO 210
 460 FOR i=1 TO 7
 470 IF d(i)>=2 THEN GO TO 500
 480 NEXT i
 490 GO TO 900
 500 FOR i=1 TO 25
 510 BEEP .02,INT (RND*35)
 520 NEXT i
 530 PRINT AT 21,1;"Am cistigat
!!          "
 540 PAUSE 80
 550 PRINT AT 21,1;"Vrei sa vezi
unde ai pierdut ?"
 560 PAUSE 0: LET r$=INKEY$: BEE
P .1,12: BEEP .1,22
 570 IF r$="d" THEN GO TO 620
```

```
 580 PRINT AT 21,1;"Alt joc (d/n
) ?                "
 590 PAUSE 0: LET r$=INKEY$: BEE
P .1,12: BEEP .1,22
 600 IF r$<>"d" THEN STOP
 610 GO TO 10
 620 PRINT AT 21,1;"O.K. - prive
ste !              "
 630 FOR i=1 TO 7
 640 IF d(i)>=2 THEN LET dist=i
 650 NEXT i
 660 FOR i=1 TO 8
 670 FOR j=1 TO 7
 680 IF a(i,j)=0 THEN GO TO 730
 690 IF j+dist>8 THEN GO TO 730
 700 IF a(i,j+dist)=0 THEN GO TO
 730
 710 PRINT AT 2*i,6+2*j; FLASH 1
;"<6>"
 720 PRINT AT 2*i,6+2*(j+dist);
FLASH 1;"<6>"
 730 NEXT j
 740 NEXT i
 760 FOR j=1 TO 8
 770 FOR i=1 TO 7
 780 IF a(i,j)=0 THEN GO TO 830
 790 IF i+dist>8 THEN GO TO 830
 800 IF a(i+dist,j)=0 THEN GO TO
 830
 810 PRINT AT 2*i,6+2*j; FLASH 1
;"<6>"
 820 PRINT AT 2*(i+dist),6+2*j;
FLASH 1;"<6>"
 830 NEXT i
 840 NEXT j
 850 GO TO 580
 900 PRINT AT 21,1;"Mutarea mea
                   "
 905 FOR i=1 TO 5
 910 DIM f(7)
 920 LET lin=INT (RND*8)+1: LET
col=INT (RND*8)+1
 930 IF a(lin.col)=1 THEN GO TO
1080
 940 FOR k=1 TO 8
 950 IF k=col THEN GO TO 1000
 960 IF a(lin,k)=0 THEN GO TO 10
00
 970 LET dist=ABS (k-col)
 980 LET f(dist)=f(dist)+1
1000 NEXT k
1010 FOR k=1 TO 8
1020 IF k=lin THEN GO TO 1060
1030 IF a(k,col)=0 THEN GO TO 10
60
1040 LET dist=ABS (k-lin)
1050 LET f(dist)=f(dist)+1
1060 NEXT k
1062 FOR t=1 TO 7
1065 IF d(t)+f(t)>=2 THEN GO TO
1080
1070 NEXT t
1075 GO TO 1310
1080 NEXT i
1081 FOR l=1 TO 8
1082 LET lin=l
1090 FOR c=1 TO 8
1092 LET col=c
1094 DIM f(7)
1100 IF a(lin,col)=1 THEN GO TO
1240
1110 FOR k=1 TO 8
1120 IF k=col THEN GO TO 1160
1130 IF a(lin,k)=0 THEN GO TO 11
60
1140 LET dist=ABS (k-col)
1150 LET f(dist)=f(dist)+1
1160 NEXT k
1170 FOR k=1 TO 8
1180 IF k=lin THEN GO TO 1220
1190 IF a(k,col)=0 THEN GO TO 12
20
1200 LET dist=ABS (k-lin)
1210 LET f(dist)=f(dist)+1
1220 NEXT k
1222 FOR t=1 TO 7
1225 IF d(t)+f(t)>=2 THEN GO TO
1240
1230 NEXT t
1235 GO TO 1310
1240 NEXT c
1250 NEXT l
1255 PRINT AT 21,1;"Ai cistigat
- felicitari.      "
1260 FOR i=1 TO 10
1270 BEEP 1,INT (RND*-10)
1280 NEXT i
1300 PAUSE 80: GO TO 580
1310 LET a(lin,col)=1
1320 PRINT AT 2*lin,6+2*col; FLA
SH 1;"<6>"
1330 PAUSE 80: PRINT AT 2*lin.6+
2*col;"<CAPS 8>"
1335 BEEP .1,12: BEEP .1,22
1340 LET juc=2: GO TO 340
```

TRIPLET

Jocul este din aceeași clasă cu **Jocul evitării pătratelor** și cu **Jocul distanțelor.** Ceea ce se urmărește de această dată este evitarea realizării de triplete, trei piese proprii aliniate orizontal, vertical sau diagonal. Poate fi folosită orice tablă caroiată (chiar dreptunghiulară), dar pe table de dimensiuni pare, cel de-al doilea jucător are strategie de cîștig, imitînd simetric față de centrul tablei mutările adversarului. De aceea, programul vă propune să jucați **Triplet** pe o tablă de dimensiuni 7×7, iar atunci cînd mută el primul, așază o piesă în centru, pentru a preveni jocul simetric față de cîmpul central.

Tabla este marcată ca la șah (litere **a—g** pentru coloane, cifre **1—7** pentru linii), o mutare fiind, deci, indicată prin coordonatele cîmpului unde dorim să așezăm o piesă (litera, apoi cifra). Programul întreabă înainte de partidă « Cine mută primul C = calculatorul/J = jucătorul)? », iar după partidă dacă se dorește « Alt joc (d/n)? ». În ceea ce privește strategia urmărită, la fel ca în cazul programelor pentru **Jocul evitării pătratelor** și **Jocul distanțelor,** el joacă mai ales « la greșeala adversarului ». Concret, el mută astfel încît să nu alinieze trei piese proprii, alegînd, însă cîmpul de joc în așa fel încît să nu consume un cîmp-capcană pentru adversar: nu joacă în linie cu două piese ale adversarului decît dacă nu are o altă posibilitate. Bineînțeles, această prevedere nu-i asigură o tărie deosebită, dar, mai ales atunci cînd joacă al doilea, nu este deloc un adversar de subestimat.

O **modificare** relativ simplă, dar care i-ar da un plus considerabil de tărie (fără a cere un timp de răspuns mult mai mare) este aceea de a juca cît mai multe mutări fără a așeza două piese proprii pe aceeași linie orizontală, verticală sau diagonală. Evident, pot fi făcute astfel cel mult șapte mutări, după care trebuie revenit la strategia din versiunea de aici.

Descrierea programului

20 — matricea **a** descrie tabla de joc (cîmpurile ocupate cu piese ale calculatorului sînt marcate cu cifra 1, iar cele ocupate cu piese ale jucătorului sînt marcate cu cifra 2); matricea **b** indică, pentru fiecare cîmp, numărul maxim de piese ale jucătorului de pe o linie care trece prin acel cîmp.

30—100 — se desenează tabla de joc și i se marchează liniile și coloanele.

110—130 — opțiunea privind prima mutare.

140—240 — se înregistrează mutarea jucătorului (în cîmpul de coordonate **lin, col**) și se verifică dacă este corectă.

250 — se înregistrează mutarea jucătorului.

260 — se efectuează mutarea jucătorului.

270 — **tl** = total pe linie, **tc** = total pe coloană.
280—310 — se numără piesele pe linia **lin** și pe coloana **col**.
320—330 — s-a realizat deja un triplet (orizontal dacă **w** = 1 și vertical dacă **w** = 2).
340—370 — se modifică matricea **b**.
380 — **td** = total piese pe o diagonală înclinată spre dreapta, **ts** = total piese pe o diagonală înclinată spre stînga.
390—435 — se numără piesele pe diagonalele care trec prin cîmpul de coordonate **lin, col**.
440—445 — s-a realizat un triplet (**w** = 3 sau **w** = =4 indică direcția lui).
450—500 — se modifică matricea **b**.
520—540 — mesaj de victorie a calculatorului, cu indicarea mutării la care jucătorul a pierdut.
550—580 — se indică direcția tripletului format.
590—620 — opțiunea de reluare.
640—650 — prima mutare a calculatorului, dacă el e primul, se face în centru.
660—880 — se caută o mutare prin care nu se pierde (totalurile pe linii și coloane, calculate la liniile 710—750, și cele pe diagonale, spre stînga și spre dreapta, calculate la liniile 770—830, nu conduc la triplete — linia 760, respectiv, 840); se începe prin explorarea cîmpurilor care sînt cel mai puțin «ocupate» de adversar, în sensul precizat de matricea **b** (pentru a nu consuma cîmpuri în care, dacă ar juca, adversarul ar pierde).
890—900 — calculatorul pierde, nu are ce muta.
910 — se înregistrează mutarea în matricea **a**.
920 — se efectuează mutarea; piesa calculatorului (un pătrat alb) este desenată în subrutina 1000—1020 («a» în modul grafic); piesa jucătorului (linia 260) este «8» cu CAPS SHIFT în modul grafic (caracterul grafic «patrat negru»).

```
  10 BORDER 1: PAPER 6: CLS : GO
SUB 1000
  20 DIM a(7,7): DIM b(7,7): DIM
s$(7)
  30 LET s$="abcdefg": LET m=0
  40 FOR i=1 TO 8
  50 PLOT 60+16*i,36: DRAW 0,112
  60 IF i=8 THEN GO TO 90
  70 PRINT AT 18-2*i,8;i
  80 PRINT AT 18,8+2*i;s$(i)
  90 PLOT 76,20+16*i: DRAW 112,0
 100 NEXT i
 110 PRINT AT 21,0;"Cine incepe
(J = juc/C = calc) ?"
 120 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 130 IF r$="c" THEN GO TO 630
 140 LET m=2: PRINT AT 21,0;"Mut
area ta (a1 - g7)            "
 150 PAUSE 0: LET q$=INKEY$: BEE
P .1,12
 160 IF q$<"a" OR q$>"g" THEN BE
EP 1,-6: GO TO 150
 170 PRINT AT 21,21;q$
 180 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 190 IF r$<"1" OR r$>"7" THEN BE
EP 1,-6: GO TO 180
 200 PRINT AT 21,22;r$: LET lin=
8-VAL r$
 210 FOR j=1 TO 7
 220 IF q$=s$(j) THEN LET col=j:
GO TO 240
 230 NEXT j
 240 IF a(lin,col)<>0 THEN BEEP
1,-6: GO TO 140
 250 LET a(lin,col)=2
 260 PRINT AT 2+2*lin,8+2*col;"<
CAPS 8>": BEEP .03,20
 270 LET tl=0: LET tc=0
 280 FOR i=1 TO 7
 290 IF a(lin,i)=2 THEN LET tl=t
l+1
 300 IF a(i,col)=2 THEN LET tc=t
c+1
 310 NEXT i
 320 IF tl=3 THEN LET w=1: GO TO
520
 330 IF tc=3 THEN LET w=2: GO TO
520
 340 FOR i=1 TO 7
 350 IF b(lin,i)<tl THEN LET b(l
in,i)=tl
 360 IF b(i,col)<tc THEN LET b(i
col)=tc
 370 NEXT i
 380 LET td=0: LET ts=0
 390 FOR i=-6 TO 6
 400 IF lin+i<1 OR lin+i>7 THEN
GO TO 435
 405 IF col+i<1 OR col+i>7 THEN
GO TO 420
 410 IF a(lin+i,col+i)=2 THEN LE
T td=td+1
 420 IF col-i<1 OR col-i>7 THEN
GO TO 435
```

```
 430 IF a(lin+i,col-i)=2 THEN LE
T ts=ts+1
 435 NEXT i
 440 IF td=3 THEN LET w=3: GO TO
 520
 445 IF ts=3 THEN LET w=4: GO TO
 520
 450 FOR i=-6 TO 6
 460 IF lin+i<1 OR lin+i>7 THEN
GO TO 500
 465 IF col+i<1 OR col+i>7 THEN
GO TO 480
 470 IF b(lin+i,col+i)<td THEN L
ET b(lin+i,col+i)=td
 480 IF col-i<1 OR col-i>7 THEN
GO TO 500
 490 IF b(lin+i,col-i)<ts THEN L
ET b(lin+i,col-i)=ts
 500 NEXT i
 510 GO TO 630
 520 PRINT AT 21,0;"Am cistigat
!!                  "
 530 FOR i=1 TO 20: BEEP .03,INT
 (RND*30): NEXT i
 540 PRINT AT 2+2*lin,8+2*col; F
LASH 1;"<6>"
 550 IF w=1 THEN PRINT AT 21,15;
"(orizontal)": GO TO 590
 560 IF w=2 THEN PRINT AT 21,15;
"(vertical)": GO TO 590
 570 IF w=3 THEN PRINT AT 21,15;
"(diagonal NW-SE)": GO TO 590
 580 IF w=4 THEN PRINT AT 21,15;
"(diagonal NE-SW)"
 590 PAUSE 150: PRINT AT 21,0;"A
lt joc (d/n) ?                 "
 600 PAUSE 0: LET q$=INKEY$: BEE
P .1,12
 610 IF q$<>"d" THEN STOP
 620 RESTORE : GO TO 10
 630 PRINT AT 21,0;"Mutarea mea
                    "
 640 IF m>1 THEN GO TO 660
 650 LET lin=4: LET col=4: GO TO
 910
 660 FOR r=0 TO 2
 670 FOR i=1 TO 7
 680 FOR j=1 TO 7
 690 IF a(i,j)<>0 THEN GO TO 860
 700 IF b(i,j)<>r THEN GO TO 860
 710 LET tl=0: LET tc=0
 720 FOR k=1 TO 7
 730 IF a(i,k)=1 THEN LET tl=tl+
1
 740 IF a(k,j)=1 THEN LET tc=tc+
1
 750 NEXT k
 760 IF tl=2 OR tc=2 THEN GO TO
860
 770 LET td=0: LET ts=0
 780 FOR k=-6 TO 6
 790 IF i+k<1 OR i+k>7 THEN GO T
O 830
 795 IF j+k<1 OR j+k>7 THEN GO T
O 810
 800 IF a(i+k,j+k)=1 THEN LET td
=td+1
 810 IF j-k<1 OR j-k>7 THEN GO T
O 830
 820 IF a(i+k,j-k)=1 THEN LET ts
=ts+1
 830 NEXT k
 840 IF td=2 OR ts=2 THEN GO TO
860
 850 LET lin=i: LET col=j: GO TO
 910
 860 NEXT j
 870 NEXT i
 880 NEXT r
 890 PRINT AT 21,0;"Ai cistigat
- nu am ce muta !!  "
 900 FOR i=1 TO 30: BEEP .04,INT
 (RND*30): NEXT i: GO TO 590
 910 LET a(lin,col)=1
 920 PRINT AT 2+2*lin,8+2*col;"<
a>"
 930 PRINT AT 21,15;s$(col);8-li
n: BEEP .02,20
 940 PAUSE 60: GO TO 140
 999 STOP
1000 FOR i=0 TO 7: READ x: POKE
USR "a"+i,x: NEXT i
1010 DATA 255,129,129,129,129,12
9,129,255
1020 RETURN
```

jocul drumului albastru

Se dau o tablă caroiată, de dimensiuni **n×n,** și trei tipuri de pătrate, ca în figura 6.

Cîmpul din stînga-sus al tablei are mijlocul laturii de sus marcat cu o săgeată. Pe rînd, cei doi jucători așază un pătrat de tip **a, b** sau **c,** la alegere, în așa fel încît drumul al cărui început este indicat de săgeată să fie prelungit (se joacă, deci, în căsuța liberă, imediat următoare capătului de drum construit pînă în acel moment). Dacă prin așezarea unui pătrat se face legătura cu un segment izolat de drum

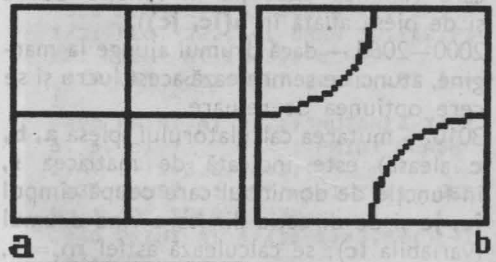


Fig. 6

de pe piesele anterior introduse în joc, drumul principal se consideră prelungit și cu acel segment (așa cum se întîmplă la mutarea 11 în partida din figura 7, care valorifică segmente de drum de pe două piese jucate anterior, la pasul 8 și, respectiv, 5).

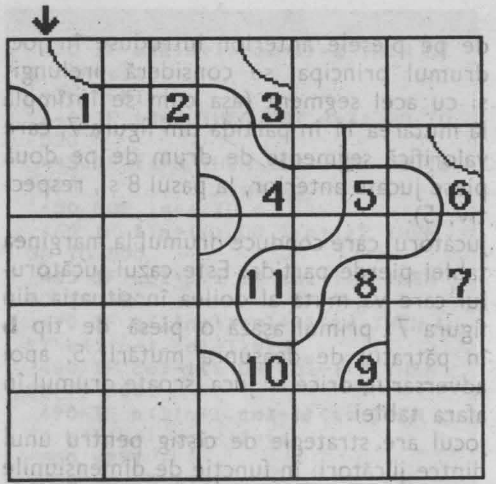
Jucătorul care conduce drumul la marginea tablei pierde partida. Este cazul jucătorului care va muta al doilea în situația din figura 7: primul așază o piesă de tip **b** în pătratul de deasupra mutării 5, apoi adversarul, orice ar juca, scoate drumul în afara tablei.

Jocul are strategie de cîștig pentru unul dintre jucători, în funcție de dimensiunile tablei. Mai exact, pe table avînd cel puțin o latură de lungime pară, poate învinge totdeauna primul jucător. Pe table cu ambele laturi impare, poate învinge totdeauna al doilea jucător. Ideea care se aplică de fiecare dată este următoarea. O tablă de suprafață pară poate fi acoperită cu dominouri de la început; o tablă cu suprafața impară, poate fi acoperită cu dominouri după efectuarea primei mutări. În ambele cazuri, în momentul împărțirii în dominouri, drumul se găsește pe o latură de domino (nu pe linia centrală a acestuia). Jucătorul aflat la mutare, joacă în așa fel încît drumul să ajungă pe linia centrală a dominoului respectiv. Adversarul nu are ce face altceva decît să conducă drumul la o nouă margine de domino. Și așa mai departe. Linia centrală a dominourilor nu se poate însă găsi pe marginea tablei; cel care face împărțirea în dominouri nu poate, deci, pierde niciodată.

Programul care urmează vă propune să jucați **Drumul albastru** pe o tablă de dimensiuni 5×5, lăsîndu-vă pe dumneavoastră să jucați primul. Și ,desigur, el cunoaște strategia descrisă mai înainte . . .

Pentru că locul unde trebuie jucat este bine precizat de fiecare dată, pentru mutare trebuie doar să indicați litera care identifică tipul de pătrat pe care doriți să-l folosiți — **a, b** sau **c,** conform figurii 6 (cele trei pătrate sînt desenate și pe ecran).

După ce ați pierdut o partidă, puteți relua jocul, desigur cu aceleași șanse, (singura performanță ar fi durata unei partide).



Programul ar putea fi **modificat** în mai multe moduri: opțiune pentru cine începe primul, table de joc de dimensiuni variabile etc. Schimbările necesare sînt însă atît de consistente încît, practic, va fi nevoie de realizarea unui program diferit.

Descrierea programului

30 — matricea **s** conține strategia de joc a programului: în funcție de linia (**ic**) și coloana (**jc**) care identifică un cîmp al tablei de joc și de direcția (**lc**) din care vine drumul, se indică 0 (nici o acțiune, situație imposibilă), 1, 2 sau 3 (mutare cu o piesă de tip **a, b,** respectiv, **c**), conform unei anumite împărțiri a tablei în dominouri; matricea **g** este folosită mai tîrziu, pentru precizarea unei ramificări în program (linia 600).
40 — 110 — citirea matricelor **s** și **g**.
120 — matricea **a** descrie tabla de joc; **p** = 1 arată că la mutare este jucătorul, **p** = 2 indică faptul că mută calculatorul.
200—260 — se desenează tabla de joc.
310 — 350 — se desenează cele trei « piese » **a, b, c,** în dreapta tablei de joc.
410 — **ic** și **jc** indică cîmpul în care trebuie jucat, iar **lc** direcția din care vine drumul construit pînă acum (inițial, de sus).
420—440 — se cere mutarea jucătorului.
450—480 — se verifică mutarea jucătorului.
500—520 — se desenează o piesă de tip **a** în cîmpul de coordonate **ic, jc.**
530—550 — se desenează o piesă de tip



Fig.7

b în cîmpul de coordonate **ic, jc.**
560—570 — se desenează o piesă de tip **c** în cîmpul de coordonate **ic, jc.**
600 — în funcție de direcția din care vine drumul și de piesa jucată (**m** = 1, 2, 3 indică piesa **a, b,** respectiv, **c**), se face salt la un alt loc din program, pentru îngroșarea drumului și prelungirea lui, dacă este cazul, cu segmente jucate anterior.
700—1840 — se îngroașă drumul pe piesa nou jucată, se schimbă valorile coordonatelor **ic** și **jc,** după cum este cazul, și se verifică dacă drumul a ajuns la marginea tablei, pe direcția corespunzătoare variabilei **ic** sau **jc** modificate; atunci cînd direcția drumului se modifică, se schimbă și valoarea variabilei **lc;** dacă în cîmpul **ic, jc** (la care a ajuns drumul) avem deja o piesă (dacă **m** = **a(ic, jc)** > 0), atunci drumul se continuă (se revine la linia 600, care ramifică execuția în funcție de **lc** și de piesa aflată în **a(ic, jc)**).
2000—2060 — dacă drumul ajunge la margine, atunci se semnalează acest lucru și se cere opțiunea de reluare.
3010 — mutarea calculatorului (piesa **a, b, c** aleasă) este indicată de matricea **s,** în funcție de dominoul care ocupă cîmpul **ic, jc** și de direcția din care vine drumul (variabila **lc**); se calculează astfel **m** = 1, 2, 3.
3020—3040 — în funcție de valoarea lui **m** (deci de piesa jucată), se merge la 500, 530 sau 560, pentru a prelungi drumul; desenarea drumului, prelungirea repetată dacă este cazul etc. se fac pentru ambii jucători în aceeași parte a programului, liniile 500—1840.

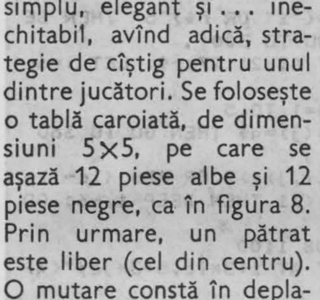
```
  10 BORDER 1: PAPER 6: INK 1
  20 PRINT FLASH 1;AT 10,4;"JOCU
L DRUMULUI ALBASTRU"
  30 DIM s(5,5,4): DIM g(4,3)
  40 FOR i=1 TO 5: FOR j=1 TO 5:
 FOR k=1 TO 4
  50 READ s(i,j,k)
  60 NEXT k: NEXT j: NEXT i
  70 DATA 0,0,0,0,0,0,2,1,0,1,3,
0,0,2,0,3,0,0,0,3,3,0,2,0,2,1,3,
0,1,2,0,3,0,3,1,2,0,0,1,2,3,0,2,
0,2,1,3,0,0,3,1,2,3,0,2,1,2,0,3,
0,1,2,0,0,1,2,0,3,3,0,2,1,2,1,3,
0,1,0,0,3,0,3,0,0,0,3,0,2,3,0,0,
1,2,1,0,0,0,0,0,2
  80 FOR i=1 TO 4: FOR j=1 TO 3
  90 READ g(i,j)
 100 NEXT j: NEXT i
 110 DATA 700,800,900,1000,1100,
1200,1300,1400,1500,1600,1700,18
00
 120 DIM a(5,5): LET p=1: CLS
 200 FOR i=1 TO 6
 210 PLOT 32,32+(i-1)*24: DRAW 1
20,0
 230 PLOT 32+(i-1)*24,32: DRAW 0
,120
 240 NEXT i
 250 PLOT 30,30: DRAW 124,0: DRA
W 0,124: DRAW -124,0: DRAW 0,-12
4
 260 PLOT 44,154: DRAW 0,12: PLO
T 44,154: DRAW 5,5: PLOT 44,154:
 DRAW -5,5
 310 FOR i=1 TO 3
 315 PLOT 210,40*i: DRAW 24,0: D
RAW 0,24: DRAW -24,0: DRAW 0,-24
 320 NEXT i
 330 PLOT 210,132: DRAW 24,0: PL
OT 222,120: DRAW 0,24: PRINT AT
5,30;"a"
 340 PLOT 210,92: DRAW 12,12,PI/
2: PLOT 222,80: DRAW 12,12,-PI/2
: PRINT AT 10,30;"b"
 350 PLOT 210,52: DRAW 12,-12,-P
I/2: PLOT 222,64: DRAW 12,-12,PI
/2: PRINT AT 15,30;"c"
 410 LET ic=1: LET jc=1: LET lc=
1
 420 PRINT AT 21,1;"MUTAREA TA (
a, b, c) ?         ": PRINT FLASH
1;AT 1+ic*3,2+jc*3;"?"
 430 PAUSE 0: LET m$=INKEY$: BEE
P .1,12: BEEP .2,22
 440 PRINT AT 4+(ic-1)*3,5+(jc-1
)*3;" "
 450 IF m$="a" THEN LET m=1: LET
 a(ic,jc)=1: GO TO 500
 460 IF m$="b" THEN LET m=2: LET
 a(ic,jc)=2: GO TO 530
 470 IF m$="c" THEN LET m=3: LET
 a(ic,jc)=3: GO TO 560
 480 PRINT AT 21,1;"MUTARE ERONA
TA            ": BEEP .5,12: GO
 TO 420
 500 PLOT 44+24*(jc-1),128-24*(i
c-1): DRAW 0,24
 510 PLOT 32+24*(jc-1),140-24*(i
c-1): DRAW 24,0
 520 GO TO 600
 530 PLOT 44+24*(jc-1),128-24*(i
c-1): DRAW 12,12,-PI/2
 540 PLOT 32+24*(jc-1),140-24*(i
c-1): DRAW 12,12,PI/2
 550 GO TO 600
 560 PLOT 32+24*(jc-1),140-24*(i
c-1): DRAW 12,-12,-PI/2
 570 PLOT 44+24*(jc-1),152-24*(i
c-1): DRAW 12,-12,PI/2
 600 GO TO g(lc,m)
 700 PLOT 43+24*(jc-1),152-24*(i
c-1): DRAW 0,-24: PLOT 45+24*(jc
-1),152-24*(ic-1): DRAW 0,-24
 705 BEEP .3,22
 710 LET ic=ic+1
 720 IF ic=6 THEN GO TO 2000
 730 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
 740 GO TO 3000
 800 PLOT 43+24*(jc-1),152-24*(i
c-1): DRAW -11,-11,-PI/2: PLOT 4
5+24*(jc-1),152-24*(ic-1): DRAW
-13,-13,-PI/2
 805 BEEP .3,22
 810 LET jc=jc-1: LET lc=2
 820 IF jc=0 THEN GO TO 2000
 830 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
 840 GO TO 3000
 900 PLOT 43+24*(jc-1),152-24*(i
c-1): DRAW 13,-13,PI/2: PLOT 45+
24*(jc-1),152-24*(ic-1): DRAW 11
,-11,PI/2
 905 BEEP .3,22
 910 LET jc=jc+1: LET lc=4
 920 IF jc=6 THEN GO TO 2000
 930 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
 940 GO TO 3000
1000 PLOT 32+24*jc,139-24*(ic-1)
: DRAW -24,0: PLOT 32+24*jc,141-
24*(ic-1): DRAW -24,0
1005 BEEP .3,22
1010 LET jc=jc-1
1020 IF jc=0 THEN GO TO 2000
1030 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
1040 GO TO 3000
1100 PLOT 32+24*jc,139-24*(ic-1)
: DRAW -11,-11,PI/2: PLOT 32+24*
jc,141-24*(ic-1): DRAW -13,-13,P
I/2
1105 BEEP .3,22
1110 LET ic=ic+1: LET lc=1
1120 IF ic=6 THEN GO TO 2000
1130 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
1140 GO TO 3000
1200 PLOT 32+24*jc,139-24*(ic-1)
: DRAW -13,13,-PI/2: PLOT 32+24*
```

```
jc,141-24*(ic-1): DRAW -11,11,-P
I/2
1205 BEEP .3,22
1210 LET ic=ic-1: LET lc=3
1220 IF ic=0 THEN GO TO 2000
1230 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
1240 GO TO 3000
1300 PLOT 43+24*(jc-1),128-24*(i
c-1): DRAW 0,24: PLOT 45+24*(jc-
1),128-24*(ic-1): DRAW 0,24
1305 BEEP .3,22
1310 LET ic=ic-1
1320 IF ic=0 THEN GO TO 2000
1330 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
1340 GO TO 3000
1400 PLOT 43+24*(jc-1),128-24*(i
c-1): DRAW 13,13,-PI/2: PLOT 45+
24*(jc-1),128-24*(ic-1): DRAW 11
,11,-PI/2
1405 BEEP .3,22
1410 LET jc=jc+1: LET lc=4
1420 IF jc=6 THEN GO TO 2000
1430 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
1440 GO TO 3000
1500 PLOT 43+24*(jc-1),128-24*(i
c-1): DRAW -11,11,PI/2: PLOT 45+
24*(jc-1),128-24*(ic-1): DRAW -1
3,13,PI/2
1505 BEEP .3,22
1510 LET jc=jc-1: LET lc=2
1520 IF jc=0 THEN GO TO 2000
1530 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
1540 GO TO 3000
1600 PLOT 32+24*(jc-1),139-24*(i
c-1): DRAW 24,0: PLOT 32+24*(jc-
1),141-24*(ic-1): DRAW 24,0
1605 BEEP .3,22
1610 LET jc=jc+1
1620 IF jc=6 THEN GO TO 2000
1630 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
1640 GO TO 3000
1700 PLOT 32+24*(jc-1),139-24*(i
c-1): DRAW 13,13,PI/2: PLOT 32+2
4*(jc-1),141-24*(ic-1): DRAW 11,
11,PI/2
1705 BEEP .3,22
1710 LET ic=ic-1: LET lc=3
1720 IF ic=0 THEN GO TO 2000
1730 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
1740 GO TO 3000
1800 PLOT 32+24*(jc-1),139-24*(i
c-1): DRAW 11,-11,-PI/2: PLOT 32
+24*(jc-1),141-24*(ic-1): DRAW 1
3,-13,-PI/2
1805 BEEP .3,22
1810 LET ic=ic+1: LET lc=1
1820 IF ic=6 THEN GO TO 2000
1830 LET m=a(ic,jc): IF m>0 THEN
 GO TO 600
1840 GO TO 3000
2000 IF p=2 THEN GO TO 2070
2005 PRINT AT 21,1;"AM INVINS -
Alt joc (d/n) ?  "
2010 FOR i=1 TO 20: BEEP .03,INT
 (RND*20)+14: NEXT i
2020 PAUSE 0: IF INKEY$<>"d" THE
N STOP
2060 GO TO 120
2070 PRINT AT 21,1;"AI INVINS -
Alt joc (d/n) ? ": GO TO 2010
3000 IF p=2 THEN LET p=1: GO TO
420
3005 LET p=2: PRINT AT 21,1;"MUT
AREA MEA (APASA O TASTA)": PAUSE
 0: BEEP .1,12: BEEP .2,22
3010 LET m=s(ic,jc,lc): LET a(ic
,jc)=m
3020 IF m=1 THEN PRINT AT 21,29;
"a": PAUSE 50: GO TO 500
3030 IF m=2 THEN PRINT AT 21,29;
"b": PAUSE 50: GO TO 530
3040 IF m=3 THEN PRINT AT 21,29;
"c": PAUSE 50: GO TO 560
```

Jocul pătratelor alunecătoare

Şi acesta este un joc din clasa **Drumului albastru** simplu, elegant şi . . . inechitabil, avînd adică, strategie de cîştig pentru unul dintre jucători. Se foloseşte o tablă caroiată, de dimensiuni 5×5, pe care se aşază 12 piese albe şi 12 piese negre, ca în figura 8. Prin urmare, un pătrat este liber (cel din centru). O mutare constă în deplasarea unei piese, din cîmpul unde se găseşte, în cîmpul liber; deplasarea se face orizontal sau vertical (nu şi diagonal). Primul trebuie, deci, să mute jucătorul cu piesele albe. Jucătorul care se află la rînd şi nu poate muta, pierde partida.

Fig.8



În program, sînteţi invitat să mutaţi primul (cu albele); o mutare este indicată prin precizarea coordonatelor piesei care se deplasează, conform notaţiei specificate de figură (ca la şah: literă—cifră). Şi nu întîmplător programul vă lasă să jucaţi primul: al doilea jucător are strategie de cîştig. Această strategie este similară celei de la **Drumul albastru:** se consideră tabla (mai puţin centrul) împărţită în dominouri; la fiecare moment, locul gol se va afla într-un domino, iar al doilea pătrat al dominoului va fi ocupat cu o piesă a programului. Acesta mută în interiorul dominoului şi astfel, la pasul următor, jucătorul va produce o configuraţie de acelaşi tip (cu un domino format din locul gol şi o piesă a calculatorului). Calculatorul are, deci, în fiecare moment posibilitatea de a muta. Împărţirea în dominouri folosită în program este cea indicată în figura 9. După . . . cîştigarea fiecărei partide, programul vă propune « Alt joc (d/n)? ».

Modificări posibile:

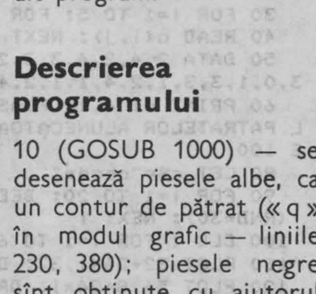
— opţiune pentru cine mută primul.
— posibilitatea de a aşeza piesele la început în altă poziţie, aleasă de jucător. Ambele modificări sînt însă de profunzime (a doua schimbă şi jocul), cerînd practic realizarea unui alt program.

Descrierea programului

10 (GOSUB 1000) — se desenează piesele albe, ca un contur de pătrat (« q » în modul grafic — liniile 230, 380); piesele negre sînt obţinute cu ajutorul semnelor grafice **Basic** (« 8 » cu CAPS SHIFT în modul grafic).

20 — matricea **s** repre-

Fig.9



zintă tabla de joc (completată cu cifre 1 în pozițiile ocupate de piese ale calculatorului și cu cifre 2 în pozițiile ocupate de piese ale jucătorului; cîmpul gol este completat cu zero); matricea **a** conține strategia de joc (direcția în care trebuie mutat, pentru fiecare cîmp gol în parte de pe tabla 5×5).

30—50—citirea matricei **a**.

100—160 — se desenează și se marchează tabla.

170—190 — se citește matricea **s** (poziția inițială).

210—240 — se desenează piesele pe tablă.

250 — **ic** și **jc** sînt coordonatele locului gol.

260—360 — se introduce și se verifică mutarea jucătorului.

370 (GOSUB 1100) — mutarea efectivă este efectuată la liniile 1100—1130.

390 — înregistrarea mutării în matricea **s**.

400 — noile coordonate ale locului gol.

410—440 — precizarea piesei care se va muta, conform strategiei înregistrate în matricea **a**.

460—470 — efectuarea mutării calculatorului (tot prin intermediul liniilor 1100—1130).

480—490 — înregistrarea mutării în matricea **s** și calcularea coordonatelor locului gol.

500—570 — se testează dacă lîngă locul gol există o piesă albă; în caz afirmativ se revine la linia 260 (se cere o nouă mutare din partea jucătorului).

580—620 — mesaj de victorie pentru calculator și opțiunea de reluare.

```
  10 PAPER 6: INK 1: BORDER 1: C
LS : RESTORE 1000: GO SUB 1000
  20 DIM a(5,5): DIM s(5,5): DIM
 c$(5)
  25 RESTORE 40
  30 FOR i=1 TO 5: FOR j=1 TO 5
  40 READ a(i,j): NEXT j: NEXT i
  50 DATA 2,4,2,4,3,3,2,4,3,1,1,
3,0,1,3,3,1,2,4,1,1,2,4,2,4
  60 PRINT AT 6,1; FLASH 1;"JOCU
L PATRATELOR ALUNECATOARE": PAUS
E 100
  80 LET c$="abcde"
  90 FOR i=1 TO 20: BEEP .02,INT
 (RND*30): NEXT i
 100 CLS : FOR i=1 TO 6
 110 PLOT 32+24*i,32: DRAW 0,120
 120 PLOT 56,8+24*i: DRAW 120,0
 130 IF i=6 THEN GO TO 160
 140 PRINT AT 19-3*i,5;i
 150 PRINT AT 19,5+3*i;c$(i)
 160 NEXT i
 170 FOR i=1 TO 5: FOR j=1 TO 5
 180 READ s(i,j): NEXT j: NEXT i
 190 DATA 1,2,1,2,1,2,1,2,1,2,1,
2,0,2,1,2,1,2,1,2,1,2,1,2,1
 200 PLOT 54,30: DRAW 124,0: DRA
W 0,124: DRAW -124,0: DRAW 0,-12
4
 210 FOR i=1 TO 5: FOR j=1 TO 5
 220 IF s(i,j)=1 THEN PRINT AT 1
+3*i,5+3*j;"<CAPS 8>"
 230 IF s(i,j)=2 THEN PRINT AT 1
+3*i,5+3*j;"<q>"
 240 NEXT j: NEXT i
 250 LET ic=3: LET jc=3
 260 PRINT AT 21,2;"Mutarea ta (
a1 - e5) =         "
 270 PAUSE 0: LET q$=INKEY$: BEE
P .1,12
```

```
 280 IF q$<"a" OR q$>"e" THEN BE
EP 1,-6: GO TO 270
 290 PRINT AT 21,26;q$
 300 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 310 IF r$<"1" OR r$>"5" THEN BE
EP 1,-6: GO TO 300
 320 PRINT AT 21,27;r$: LET i=6-
VAL r$
 330 FOR j=1 TO 5
 340 IF c$(j)=q$ THEN GO TO 360
 350 NEXT j
 360 IF s(i,j)<>2 OR ABS (ic-i)+
ABS (jc-j)<>1 THEN BEEP 1,-6: GO
 TO 260
 370 GO SUB 1100
 380 PRINT AT 1+3*ic,5+3*jc;"<q>
"
 390 LET s(i,j)=0: LET s(ic,jc)=
2
 400 LET ic=i: LET jc=j
 410 IF a(ic,jc)=1 THEN LET i=ic
-1: GO TO 450
 420 IF a(ic,jc)=2 THEN LET j=jc
+1: GO TO 450
 430 IF a(ic,jc)=3 THEN LET i=ic
+1: GO TO 450
 440 LET j=jc-1
 450 PRINT AT 21,2;"Mutarea mea
(apasa o tasta)"
 460 PAUSE 0: BEEP .1,12: GO SUB
 1100
 470 PRINT AT 1+3*ic,5+3*jc;"<CA
PS 8>"
 480 LET s(i,j)=1: LET s(ic,jc)=
1
 490 LET ic=i: LET jc=j
 500 IF ic=1 THEN GO TO 520
 510 IF s(ic-1,jc)=2 THEN GO TO
260
```

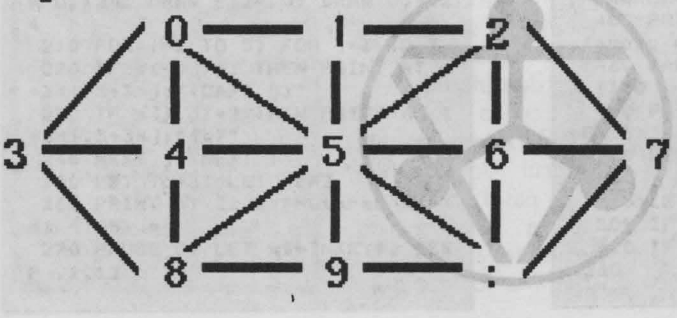
```
520 IF jc=5 THEN GO TO 540
530 IF s(ic,jc+1)=2 THEN GO TO
260
540 IF ic=5 THEN GO TO 560
550 IF s(ic+1,jc)=2 THEN GO TO
260
560 IF jc=1 THEN GO TO 580
570 IF s(ic,jc-1)=2 THEN GO TO
260
580 PRINT AT 21,2;"Am cistigat
! Alt joc (d/n) ?"
590 FOR i=1 TO 20: BEEP .03,INT
(RND*30): NEXT i
600 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
```

```
610 IF r$<>"d" THEN STOP
620 RESTORE 190: GO TO 100
999 STOP
1000 FOR i=0 TO 7: READ x: POKE
USR "q"+i,x: NEXT i
1010 DATA 255,129,129,129,129,12
9,129,255
1020 RETURN
1100 PRINT AT 1+3*i,5+3*j; FLASH
 1;"(6)": PAUSE 100
1110 PRINT AT 1+3*i,5+3*j;" "
1120 PRINT AT 1+3*ic,5+3*jc; FLA
SH 1;"(6)": PAUSE 100
1130 RETURN
```

vînătoare engleză

Pe ecran apare o rețea (tablă de joc) cu 11 noduri. Fiecare nod din primele 10 este simbolizat prin cifre de la **0** la **9,** iar al 11-lea nod prin simbolul « : », conform schemei afișate pe ecran (fig.10). Pe nodurile rețelei sînt poziționați 3 copoi (reprezentanți grafic prin litera C) și un iepure (reprezentat grafic prin litera I). Un copoi se poate deplasa în alt nod dacă există traseu între nodurile respective, scopul jocului fiind de a prinde iepurele într-un nod de unde nu se mai poate deplasa.

Fig.10



Jocul este asemănător cu cel numit « moara », joc ce se juca — la moară — cu un bob de grîu și trei de porumb, un iepure și trei cîini.

Jocul se desfășoară astfel: jucătorul va indica o mutare pentru deplasarea unui copoi, după care calculatorul va efectua o mutare a iepurelui. Indicarea unei mutări de către jucător se va face astfel: se tastează simbolul nodului unde se află copoiul care se dorește a se deplasa și simbolul nodului unde se dorește să ajungă. Apoi se va acționa tasta CR(ENTER). Pentru indicarea nodului al 11-lea se va acționa tasta Z împreună cu SYMBOL SHIFT. De exemplu, pentru deplasarea unui copoi din nodul **5** în nodul :, se va tasta 5 și apoi SYMBOL SHIFT + Z.

Există 3 grade de dificultate, diferențiate prin poziția inițială. Stabilirea nivelului se face de jucător la începutul jocului (în urma întrebării adresate de calculator). Pentru a prinde iepurele, jucătorul are la dispoziție 15 mutări, după care se poate începe, la dorința jucătorului, un joc nou. Pe ecran apar afișate permanent numărul de mutări efectuate, precum și numărul de mutări rămase. Poziția importantă a jocului se poate observa în fig. 11, bineînțeles existînd și poziția simetrică (cu cîinele în nodul **7**). În ambele poziții trebuie găsită o mutare, astfel încît iepurele să nu aibă altă alternativă decît nodul **4** (de unde va avea doar posibilitatea reîntoarcerii în nodul **3**) și, în același timp, să existe și pentru copoi posibilitatea de mutare. Astfel, mutarea copoiului din nodul **1** în nodul **0** nu este bună, deoarece, după forțarea iepurelui la nodul **4**, copoii nu vor mai avea o replică fără a lăsa alternativă de scăparea iepurelui. Mutarea corectă este de la nodul **5** la nodul **0,** după care urmarea este foarte simplă. Din orice altă poziție de joc (inclusiv cele cu grade de dificultate sporită) se poate ajunge în poziția amintită.

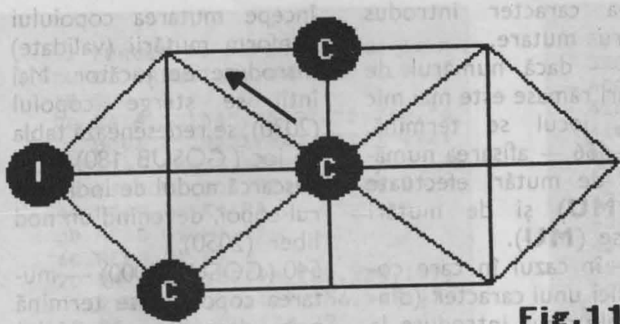


Fig. 11

Descrierea programului

20 — date pentru citirea variabilelor tip șir de caractere folosite pentru strategia de joc a iepurelui.
30—60 — citirea variabilelor tip șir de caractere (**A$** — **K$**) folosite pentru strategia de joc a iepurelui. **A$** reprezintă strategia iepurelui atunci cînd acesta se află în nodul **0** (din nodul **0** se poate deplasa într-unul din nodurile **5, 1, 4** sau **3**). Similar se vor forma perechi **B$** — iepure în nod **1, C$** —iepure în nod **2** etc. Cînd iepurele se află într-un nod și trebuie să efectueze o mutare, se vor inspecta pe rînd nodurile, conform informațiilor conținute în variabila tip șir de caractere asociată, iepurele mutîndu-se în primul nod liber găsit astfel.
70 — date reprezentînd coordonatele punctelor — noduri ale tablei de joc de pe ecranul grafic.
80 — rezervarea de spații de memorie pentru matricea **K** în elementele căreia sînt memorate coordonatele de puncte ale tablei de joc.
90—130 — citirea coordonatelor punctelor pentru tabla de joc. **K(I,1)** reprezintă coordonatele sau modificările coordonatelor de pe orizontală; **K(I,2)** reprezintă coordonatele sau modificările coordonatelor de pe verticală.
145 — inițializarea variabilei **MU,** care reprezintă numărul de mutări rămase. La începutul jocului, numărul de mutări (rămase) la dispoziția jucătorului este 15.
177 (GOSUB 180) — apelarea subrutinei de desenare a tablei de joc. 180—310 — desenarea rețelei prin intermediul coordonatelor și modificărilor de coordonate **K(I,J).**
330 — rezervarea unui spațiu de memorie variabilei — indicator **P.** Această variabilă indică starea fiecăruia din cele 11 noduri ale rețelei la un moment dat, astfel:
P(X) = 0 nodul **X** este liber;
P(X) = 1 în nodul **X** se află un copoi;
P(X) = 2 în nodul **X** se află iepurele.
350—360 — introducerea gradului de dificultate (variabila **GD**) și respingerea valorilor neinteresante pentru gradul de dificultate.
365 — ștergerea mesajului apărut pe linia de jos a ecranului.
366 — stabilirea locului de pornire a jocului (numărul liniei de program) pentru diferite grade de dificultate introduse.
370 — începerea jocului pentru gradul de dificultate 2.
370—410 — desenarea copoilor pe tabla de joc.
380 — alegerea întîmplătoare a unui număr care va reprezenta un nod de rețea în care se va pune un copoi.
390 — dacă în nodul ales se află deja un copoi, atunci se va alege altul.
400 (GOSUB 1000) — apelarea subrutinei pentru stabilirea coordonatelor în scopul desenării copoilor.
A — variabila pentru stabilirea coordonatelor pe orizontală.
B — variabila pentru stabilirea coordonatei pe verticală.
AA — variabila pentru stabilirea coloanei pe care va fi desenat caracterul corespunzător pentru copoi (C).
BB — variabila pentru stabilirea liniei pe care va fi desenat caracterul corespunzător pentru copoi. Calculul se va face conform următorului algoritm: dacă nodul rețelei (punctul) în jurul căruia se va înscrie caracterul corespunzător pentru copoi este de coordonate **(A,B),** atunci acel caracter se va înscrie în celula-caracter a cărei linie

va fi **(175-B)/8** şi a cărei coloană va fi **A/8.**
1030 — desenare copoi.
1050 — marcarea nodului de reţea cu indicatorul de copoi.
413 — alegere nod pentru iepure.
416 — dacă în nodul ales se găseşte un copoi, atunci se alege alt nod.
419 (GOSUB 3000) — apelarea subrutinei pentru stabilirea coordonatelor în scopul desenării iepurelui. Calculul se va face conform aceluiaşi algoritm ca şi cel descris pentru înscrierea caracterului copoi.
3030 — desenare iepure.
3050 — marcarea nodului de reţea cu indicatorul de iepure.
425 — început joc pentru grad de dificultate 1 sau 3.
425—427 — stabilire noduri pentru copoi (pentru gradul de dificultate 1 sau 3, nodurile pentru copoi vor fi **1, 5** şi **9**).
428 — dacă gradul de dificultate este 1, atunci iepurele se va desena în nodul **0.** Urmează acelaşi algoritm pentru înscrierea iepurelui în reţea şi marcarea nodului cu indicatorul iepure.
429 — dacă gradul de dificultate este 3, atunci iepurele se va desena într-unul din nodurile **3** sau **7.**
430 — introducere mutarea (subrutina 2500). Şirul de două caractere introduse se va păstra în variabila **MS,** care va reprezenta, deci, mutarea efectuată de jucător.
440 — **X** reprezintă primul caracter introdus pentru mutare.
450 — **Y** reprezintă al doilea caracter introdus pentru mutare.
462 — dacă numărul de mutări rămase este mai mic ca **0,** jocul se termină.
465—466 — afişarea numărului de mutări efectuate (15—**MU**) şi de mutări rămase (**MU**).
470 — în cazul în care codul nici unui caracter (dintre cele două introduse la mutare) nu reprezintă un nod, atunci introducerea mutării se va repeta (numărul de mutări rămase scăzînd însă cu o unitate).
480 — dacă la efectuarea mutării, primul caracter introdus reprezintă un nod liber sau un nod ocupat de un iepure, înseamnă că mutarea este greşită (numai copoii se pot muta de către jucător, deci, primul caracter introdus trebuie să reprezinte un nod în care se află un copoi). Mutarea se va repeta (numărul de mutări rămase scăzînd însă cu o unitate).
490 — al doilea caracter introdus reprezintă un nod ocupat.
500 (GOSUB 5000) — apelarea subrutinei pentru mutarea iepurelui.
GOSUB 7000 — apelarea subrutinei pentru salvarea nodului în care se află iepurele (în variabila **V**) şi a posibilităţilor de mutare a iepurelui (în variabila **V$**), conform nodului în care se află.
510 — dacă al doilea caracter introdus pentru mutare este identic cu primul **(X=Y),** atunci introducerea mutării se va repeta (numărul de mutări scăzînd cu o unitate).
520 (GOSUB 2000) — se începe mutarea copoiului conform mutării (validate) introduse de jucător. Mai întîi se şterge copoiul (2030), se redesenează tabla de joc (GOSUB 180) şi se descarcă nodul de indicatorul copoi, devenind un nod liber (2050).
540 (GOSUB 1000) — mutarea copoiului se termină prin desenarea copoiului pe nodul indicat de mutarea jucătorului.
580 (GOSUB 6000) — apelarea subrutinei pentru găsirea unei mutări pentru iepure.
6010—6060 — se inspectează fiecare posibilitate de mutare a iepurelui indicată da variabila tip şir de caractere **Z$.**
6050 — dacă s-a găsit un nod liber, se începe mutarea iepurelui în acel nod.
6060 — dacă s-a inspectat toată lista de posibilităţi de mutare a iepurelui şi nu s-a găsit nici un nod liber, înseamnă că iepurele este prins.
6070 — mesaj şi efecte sonore care indică prinderea iepurelui.
6090—6131 — alegere opţiune joc nou.
590 (GOSUB 4000) — apelarea subrutinei pentru iepure. Mai întîi iepurele este şters din vechea poziţie (4030), se redesenează tabla de joc (GOSUB 180) şi se descarcă nodul de indicatorul iepure, devenind un nod liber (4050).
610 (GOSUB 300) — mutarea iepurelui se termină prin desenarea iepurelui în nodul liber găsit.
620 — se reia algoritmul prin cererea de introducere a unei noi mutări.

```
   2 BORDER 7: PAPER 7: INK 0: F
LASH 0: BRIGHT 0: OVER 0: CLS
  18 CLS
  20 DATA "5134","502","5176","0
84","5083","08:21946","52:7","2:
6","5934","58:","5976"
  30 RESTORE 20: READ A$,B$,C$
  40 READ D$,E$,F$
  50 READ G$,H$,I$
  60 READ J$,K$
  70 DATA 56,159,96,159,136,159,
16,119,56,119,96,119,136,119,176
,119,56,79,96,79,136,79
  80 DIM K(11,2)
  90 FOR I=1 TO 11
 100 FOR J=1 TO 2
 110 READ K(I,J)
 120 NEXT J
 130 NEXT I
 140 CLS
 145 LET MU=15
 150 PRINT AT 9,25;"0-1-2";AT 10
,24;"/|\|/|\";AT 11,23;"3-4-5-6-
7";AT 12,24;"\|/|\|/";AT 13,25;"
8-9-:"
 165 PRINT AT 15,3;"Mutari efect
uate=  ;ramase=";
 177 GO SUB 180: GO TO 330
 180 PLOT K(4,1)+4,K(4,2)-4
 190 DRAW K(8,1)-K(4,1),K(8,2)-K
(4,2)
 200 DRAW K(3,1)-K(8,1),K(3,2)-K
(8,2)
 210 DRAW K(9,1)-K(3,1),K(9,2)-K
(3,2)
 220 DRAW K(11,1)-K(9,1),K(11,2)
-K(9,2)
 230 DRAW K(3,1)-K(11,1),K(3,2)-
K(11,2)
 240 DRAW K(1,1)-K(3,1),K(1,2)-K
(3,2)
 250 DRAW K(9,1)-K(1,1),K(9,2)-K
(1,2)
 260 DRAW K(4,1)-K(9,1),K(4,2)-K
(9,2)
 270 DRAW K(1,1)-K(4,1),K(1,2)-K
(4,2)
 280 DRAW K(11,1)-K(1,1),K(11,2)
-K(1,2)
 290 DRAW K(8,1)-K(11,1),K(8,2)-
K(11,2)
 300 PLOT K(2,1)+4,K(2,2)-4
 310 DRAW K(10,1)-K(2,1),K(10,2)
-K(2,2)
 320 RETURN
 330 DIM P(11)
 350 PRINT #1;"GRADUL DE DIFICUL
TATE (1,2,3) ?";
 360 PAUSE 0: LET GD=CODE INKEY$
-48: IF GD<1 OR GD>3 THEN GO TO
350
 365 INPUT 1
 366 IF GD<>2 THEN GO TO 425
 370 FOR Q=1 TO 3
 380 LET X=INT (RND*11)
 390 IF P(X+1)=1 THEN GO TO 380
 400 GO SUB 1000
 410 NEXT Q
 413 LET X=INT (RND*11)
 416 IF P(X+1)=1 THEN GO TO 413
 419 GO SUB 3000
 422 GO TO 430
 425 LET X=1: GO SUB 1000
 426 LET X=5: GO SUB 1000
 427 LET X=9: GO SUB 1000
 428 IF GD=1 THEN LET X=0: GO SU
B 3000
 429 IF GD=3 THEN LET X=3+4*(INT
 (RND*2)): GO SUB 3000
 430 PRINT AT 21,3;"INTRODU MUTA
REA :    ";: INPUT M$: GO SUB 250
0
 440 LET X=CODE (M$(1))-48
 450 LET Y=CODE (M$(2))-48
 460 LET MU=MU-1
 461 IF MU>-1 THEN GO TO 465
 462 PRINT AT 18,0;"   Incearca
  cu un iepure                ma
i obosit!"
 463 GO TO 6080
 465 PRINT AT 15,20;"  ";AT 15,2
0;15-MU
 466 PRINT AT 15,30;"  ";AT 15,3
0;MU
 470 IF X<0 OR X>10 OR Y<0 OR Y>
10 THEN GO TO 430
 480 IF P(X+1)=0 OR P(X+1)=2 THE
N GO TO 1500
 490 IF P(Y+1)=1 OR P(Y+1)=2 THE
N GO TO 1500
 500 GO SUB 5000
 510 IF X=Y THEN GO TO 430
 520 GO SUB 2000
 530 LET X=Y
 540 GO SUB 1000
 550 LET X=-1
 560 LET X=X+1
 570 IF NOT P(X+1)=2 THEN GO TO
560
 580 GO SUB 6000
 590 GO SUB 4000
 600 LET X=Y
 610 GO SUB 3000
 620 GO TO 430
 630 STOP
1010 LET A=K(X+1,1): LET AA=A/8
1020 LET B=K(X+1,2): LET BB=175-
B: LET BB=BB/8
1030 BEEP .1,10: PRINT AT BB,AA;
 FLASH 1;"C"
1050 LET P(X+1)=1
1060 RETURN
1500 PRINT AT 20,0;"MUTARE GRESI
TA !": PAUSE 0: PRINT AT 20,0;TA
B 20: GO TO 430
2010 LET A=K(X+1,1): LET AA=A/8
2020 LET B=K(X+1,2): LET BB=175-
B: LET BB=BB/8
```

```
2030 PRINT AT BB,AA;" "
2040 GO SUB 180
2050 LET P(X+1)=0
2060 RETURN
2500 LET M$=M$+"  ": PRINT AT 21
,21; INK 2; PAPER 6;M$(1 TO 2):
PAPER 7: INK 0: RETURN
3010 LET A=K(X+1,1): LET AA=A/8
3020 LET B=K(X+1,2): LET BB=175-
B: LET BB=BB/8
3030 BEEP .1,10: PRINT AT BB,AA;
 INVERSE 1;"I"
3050 LET P(X+1)=2
3060 RETURN
4010 LET A=K(X+1,1): LET AA=A/8
4020 LET B=K(X+1,2): LET BB=175-
B: LET BB=BB/8
4030 PRINT AT BB,AA;" "
4040 GO SUB 180
4050 LET P(X+1)=0
4060 RETURN
5000 GO SUB 7000: LET Z$=V$
5010 LET L=LEN (Z$)
5020 LET I=0
5030 LET I=I+1
5040 LET Z=CODE (Z$(I))-48
5050 IF Y=Z THEN RETURN
5060 IF I=L THEN LET Y=X: RETURN

5061 GO TO 5030
6000 GO SUB 7000: LET Z$=V$
6010 LET L=LEN (Z$)
6020 LET I=0
6030 LET I=I+1
6040 LET Z=CODE (Z$(I))-48
6050 IF P(Z+1)=0 THEN GO TO 6140
6060 IF I<L THEN GO TO 6030
6070 PRINT AT 18,5;"Iepurele est
e prins.": FOR G=1 TO 20: BEEP .
05,G: BEEP .05,G-3: NEXT G
6080 PRINT AT 20,6; FLASH 1;"JOC
UL S-A TERMINAT"
6090 INPUT 2: PRINT #2;"      MAI
 DORESTI ?(DA/NU)"
6100 PAUSE 0
6105 LET R$=INKEY$
6110 IF R$="D" OR INKEY$="d" THE
N GO TO 140
6130 CLS : PRINT AT 10,5;"CRED C
A TI-A PLACUT.": STOP
6140 IF NOT (Z=1 OR Z=9) THEN LE
T Y=Z: RETURN
6150 IF NOT (P(1)+P(6)+P(3)=4 OR
 P(9)+P(6)+P(11)=4) THEN LET Y=Z
: RETURN
6160 IF P(1)=2 OR P(9)=2 THEN GO
 TO 6190
6170 IF P(8)=0 THEN LET Y=7
6175 LET Y=6
6180 RETURN
6190 IF P(4)=0 THEN LET Y=3
6195 LET Y=4
6200 RETURN
6210 IF NOT (P(9) AND P(6) AND P
(3)=1 OR P(1) AND P(6) AND P(11)
=1) THEN LET Y=Z: RETURN
6220 IF X=4 OR X=6 THEN LET Y=Z:
 RETURN
6230 IF X=0 OR X=8 THEN LET Y=4:
 RETURN
6240 IF X=2 OR X=10 THEN LET Y=6
: RETURN
6250 LET Y=Z: RETURN
7000 LET V=X+1
7010 IF V=1 THEN LET V$=A$
7020 IF V=2 THEN LET V$=B$
7030 IF V=3 THEN LET V$=C$
7040 IF V=4 THEN LET V$=D$
7050 IF V=5 THEN LET V$=E$
7060 IF V=6 THEN LET V$=F$
7070 IF V=7 THEN LET V$=G$
7080 IF V=8 THEN LET V$=H$
7090 IF V=9 THEN LET V$=I$
7100 IF V=10 THEN LET V$=J$
7110 IF V=11 THEN LET V$=K$
7120 RETURN
9999 SAVE "vinatoare": VERIFY ""
```

REVERSI

Se joacă pe o tablă de 8×8 poziții (căsuțe), fiecare aflîndu-se la intersecția unei coloane cu o linie. În versiunea prezentată, jocul se poate desfășura între calculator și jucător sau între doi jucători, existînd și posibilitatea unei partide demonstrative simulată de calculator (O jucători).

În joc există 64 de piese, fiecare putînd fi albă sau neagră, de unde și denumirea jocului, care inițial se juca cu piese colorate în negru pe o parte și în alb pe cealaltă parte (revers).

La începutul jocului, din poziția inițială (vezi fig. 12), jucătorii își aleg culorile, piesele negre considerîndu-se a fi ale unuia, iar cele albe ale celuilalt. Jucătorul la mutare va pune o piesă de culoare aleasă inițial pe o poziție care formează o linie orizontală, verticală sau diagonală cu o poziție pe care este situată o piesă de aceeași culoare, astfel încît, toate piesele de pe aceste linii vor căpăta culoarea celor din extremități. De exemplu: în situația din fig. 13 dacă jucătorul care are piesele negre va pune o piesă

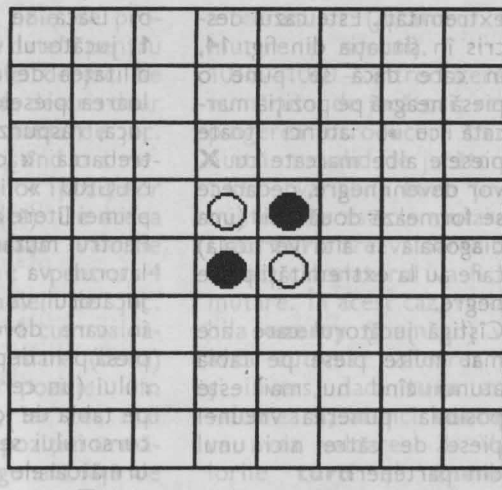
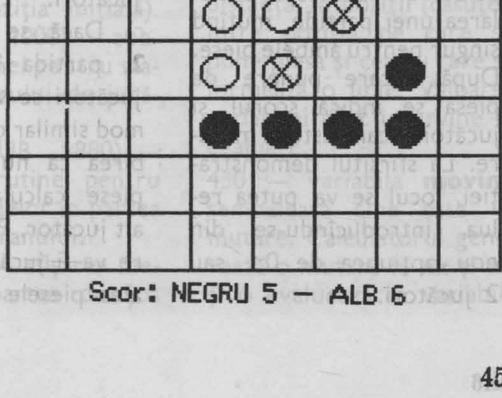


Fig.12

pe poziția marcată cu *, atunci toate piesele albe marcate cu **X** vor deveni negre.

Dacă la punerea piesei se formează mai multe linii cu poziții pe care sînt situate piese de aceeași culoare, astfel încît între acestea să existe numai poziții ocupate cu piese de culoare inversă, atunci toate piesele de pe aceste linii vor căpăta culoarea celor din

Fig.13



Scor: NEGRU 5 — ALB 6

extremități. Este cazul descris în situația din fig. 14, în care dacă se pune o piesă neagră pe poziţia marcată cu * atunci toate piesele albe marcate cu **X** vor deveni negre, deoarece se formează două linii (una diagonală și alta verticală) care au la extremități piese negre.

Cîștigă jucătorul care are mai multe piese pe tablă atunci cînd nu mai este posibilă punerea vreunei piese de către nici unul din parteneri.

Jocul prezentat începe cu întrebarea dacă se dorește sunet sau nu în timpul desfășurării partidei. Se va răspunde acționîndu-se una din tastele **D** sau **N**, după cum se dorește sau nu prezența sunetelor.

Apoi, va apărea pe ecran poziția inițială a celor 4 piese și tabla de joc. Va urma întrebarea referitoare la numărul de jucători. Se va răspunde cu **0, 1** sau **2** după cum se dorește o demonstrație de partidă (0 jucători — se va juca calculator — calculator), o partidă calculator-jucător (1 jucător) sau, respectiv, o partidă între 2 jucători.

a) Dacă se răspunde cu **0,** se va afișa scorul inițial: NEGRU 2 — ALB 2 și calculatorul va începe simularea unei partide, mutînd singur pentru ambele piese. După fiecare punere de piesă se indică scorul și jucătorul care este la mutare. La sfîrșitul demonstrației, jocul se va putea relua, introducîndu-se din nou opțiunea de 0,1 sau 2 jucători.

b) Dacă se răspunde cu **1,** jucătorul va avea posibilitatea de a-și alege culoarea pieselor cu care va juca, răspunzîndu-se la întrebarea «joci ALB sau NEGRU?» prin tastarea primei litere a culorii alese. Pentru mutarea sa, calculatorul va muta singur. Jucătorul va indica poziția în care dorește să pună piesa prin deplasarea cursorului (un cerculeț clipitor) pe tabla de joc. Deplasarea cursorului se realizează cu următoarele taste: **A** — sus, **Z** — jos, **X** — stînga, **C** — dreapta. Dacă prin aceste deplasări s-a ajuns într-o poziție pe care dorește să pună o piesă, atunci se face confirmarea mutării prin acționarea tastei **V.** Dacă se face o confirmare de mutare (se acționează tasta **V)** cînd cursorul nu este pe o poziție validă (nu se formează o linie care are la extremități piese de culoarea aleasă de jucător), mutarea (punerea piesei) nu va fi luată în considerație. După fiecare punere de piesă se indică scorul și jucătorul (NEGRU sau ALB) care este la mutare. La sfîrșitul partidei jocul se va relua, introducîndu-se din nou opțiunea de **0, 1** sau **2** jucători.

c) Dacă se răspunde cu **2,** partida între cei doi jucători se va desfășura în mod similar cu b , cu deosebirea că nu va mai pune piese calculatorul, ci un alt jucător. Primul la mutare va fi jucătorul care și-a ales piesele negre.

Descrierea programului

Pentru înțelegerea modului de realizare a programului, este necesară o descriere a algoritmului (strategiei) de punere a piesei de către calculator, aceasta fiind problema cea mai interesantă în joc.

Toate piesele de pe tablă formează o mulțime conexă. În vederea punerii piesei de către calculator, programul pornește dinspre partea de nord a formației de piese și inspectează toate pozițiile în care se poate pune o piesă, înconjurînd formația prin vest (face o tură). Pentru fiecare poziție posibilă, evaluează situația (ce s-ar întîmpla dacă ar pune acolo o piesă), obținînd un anumit punctaj, astfel: pentru pozițiile ocupate se adună punctele celui care efectuează mutarea și se scad cele ale adversarului. Fiecare poziție de pe tablă are asociată în acest algoritm o anumită valoare, formîndu-se un cîmp de valori al tuturor pozițiilor. Astfel, pozițiile din colțurile tablei au valoarea cea mai mare (100), valori mari avînd și pozițiile de pe marginea tablei.

Unele poziţii au asociate valori foarte mici sau chiar negative: acestea sînt pozițiile în care nu este indicată punerea piesei și sînt situate pe a doua linie și/sau a doua coloană a tablei de joc. Calculatorul va pune piesa pe poziția al cărei punctaj calculat, în urma evaluării,

este maxim. Evaluările pe poziții se vor putea viziona pe ecran în partea stîngă a tablei de joc sub forma de mici puncte. Aceste puncte apar pe ecran de la stînga la dreapta, urmărind ordinea de parcurs a zonelor de poziții de pe lîngă formația de piese. Locului în care punctele formează o grămadă (linie) mai înaltă îi va corespunde o zonă a tablei în care se află poziția cu punctaj maxim.

Descrierea programului

10 — rezervarea de spațiu de memorie pentru variabilele **b** — ține minte situația de pe tabla de joc — și **t** — ține minte punctajul.

30 — variabila **legal** este un indicator al legalității mutării: dacă mutarea (punerea de piesă) se poate efectua, atunci **legal** are valoarea **1**, iar dacă nu, are valoarea **0.** Variabilele șir de caractere **i$** și **j$** servesc la memorarea mesajelor care apar în partea de jos a ecranului (referitor la scor și cine este la mutare).

60(GOSUB 6000) — apelarea subrutinei pentru tactici.

610—670 — citirea datelor de descriere a tablei de joc care vor fi memorate în variabila **tactic.** În funcție de valorile pe care le ia variabila **tactic,** se vor calcula și valorile pentru variabilele **t,** care exprimă valorile asociate pozițiilor tablei de joc.

690—720 — datele asociate pozițiilor tablei de joc. Fiecare linie de program conține datele pentru o linie a tablei de joc. Se observă că este descris doar un sfert din tabla de joc. (16 date), existînd o simetrie a valorilor pozițiilor (tablă simetrică). De aceea se și calculează valorile variabilelor **t** pentru 4 zone ale tablei de joc. Se mai observă cum valoarea cea mai mare (bună) este atașată poziției din colțul tablei, iar cea mai mică (slabă), poziției următoare pe diagonală față de aceasta (intersecția liniei și coloanei 2 a tablei de joc). Dacă se modifică aceste date se va schimba și strategia de joc a programului (calculatorului).

90 — introducerea comenzii referitoare la sunet.

1000 — dacă nu a fost o comandă validă, se va repeta întrebarea solicitîndu-se din nou introducerea comenzii.

110(GOSUB 750) — apelarea subrutinei pentru desenarea poziției inițiale.

700 — inițializarea variabilelor pentru punerea pieselor. 760 — inițializarea valorii pentru înălțimea sunetelor.

770—840 — desenarea tablei de joc.

850—930 — desenarea pieselor (în poziția inițială).

120(GOSUB 1000) - apelarea subrutinei pentru stabilirea numărului de jucători.

1000 (GOSUB 1880) — apelarea subrutinei pentru afișarea mesajelor în partea de jos a ecranului.

1880—1900 — afișarea mesajelor în partea de jos a ecranului (jucătorul la mutare și scorul).

1010—1020 — introducerea numărului de jucători; respingerea introducerii unui număr nevalid de jucători.

130 — variabila **turn** ține minte cine este la mutare. Dacă **turn** are valoarea **3,** atunci calculatorul va fi la mutare. În acest caz, variabila **enemy** (inamicul) va avea automat valoarea **2** și invers, dacă **turn** are valoarea **2,** atunci **enemy** va avea valoarea **3.** Valorile **turn** și **enemy** lucrează în pereche.

140 — formarea mesajelor.

150 — apelarea subrutinei de desenare a tablei de joc și a celei de control a legalității mutării (GOSUB 1530).

180 — variabila **gamov** indică existența unei mutări posibile.

330—580 — subrutină de calcul a evaluării unei mutări.

330 — valoarea **0** a variabilei **take** indică inexistența unei mutări, dacă **legal** = **0.** La începutul evaluării se face **take = 0** și **legal = 0,** iar după evaluarea lor, ori amîndouă sînt **0**, ori amîndouă sînt **1.** (Variabilele **legal** și **take** lucrează în pereche).

Variabila **dist** reprezintă distanța, în poziții (căsuțe), între poziția pe care se pune piesa și cea cu care se formează o linie. Valoarea variabilei **dist** intervine în evaluare.

430 — variabila **moving** semnalează cine este la mutare. Calculatorul generează o mutare la **nx** și **ny** și o evaluează în variabila

scre (linia 1220), care ține minte punctajul și a cărei valoare este calculată în funcție de **t.** Dintre toate mutările posibile, calculatorul o alege pe aceea care pentru **scre** are valoarea cea mai mare, iar acest lucru se face prin intermediul variabilelor **xbest** și **ybest.**

410 — dacă **dist** = **1**, nu se merge în această direcție.

920 — desenează o piesă albă (GOSUB 1650).

930 — desenează o piesă neagră (GOSUB 1760).

1050 — calculatorul joacă cu amîndouă piesele.

1070 — joacă doi jucători.

1150 — începe efectuarea mutării de către calculator. Variabila **pts** reprezintă numărul de puncte. Se pornește inițial de la cea mai slabă situație (—1000). Variabila **pts** memorează valoarea cea mai mare a punctajului pentru fiecare evaluare (care se face cu variabila **scraux).**

1220 — evaluarea a fost terminată.

1340—1510 — subrutină de introducere a mutării jucătorului.

1390—1430 — posibilitatea de plasare a cursorului de către jucător cu tastele **A, Z, X** și **C.** Validarea mutării (cu tasta **V).**

```
  10 DIM b(8,8): DIM t(8,8): DIM
s$(26)
  20 PRINT
  30 LET legal=1: LET i$="": LET
j$=""
  40 PAPER 0: BORDER 0: INK 6: O
VER 0: FLASH 0: BRIGHT 1: BEEP 0
.01,10: CLS
  50 PRINT "   R E V E R S I"
  60 PRINT : GO SUB 600
  70 PRINT "Doriti sunet ?"
  80 PAUSE 0: LET q$=INKEY$
  90 IF q$="d" OR q$="D" OR q$="
n" OR q$="N" THEN GO TO 110
 100 GO TO 70
 110 GO SUB 750
 120 GO SUB 1000
 130 LET turn=3: LET enemy=2
 140 LET m$="Scor: NEGRU "+STR$
bp+"- ALB "+STR$ wp: GO SUB 1880
 150 GO SUB 770: GO SUB 1530: IF
 legal=0 THEN LET m$="NEGRUL nu
are mutare": GO SUB 1880: GO TO
190
 160 LET m$="mutarea NEGRULUI":
GO SUB 1880
 170 IF black=1 THEN GO SUB 1150
: GO TO 180
 175 IF black=2 THEN GO SUB 1340
 180 IF gamov=1 THEN GO TO 240
 190 LET turn=2: LET enemy=3
 200 GO SUB 770: LET m$="Scor: N
EGRU "+STR$ bp+"- ALB "+STR$ wp:
 GO SUB 1880
 210 GO SUB 1530: IF legal=0 THE
N LET m$="ALBUL nu are mutare":
GO SUB 1880: GO TO 240
 220 LET m$="mutarea ALBULUI": G
O SUB 1880
 230 IF white=1 THEN GO SUB 1150
: GO TO 240
 235 IF white=2 THEN GO SUB 1340
 240 IF gamov=0 THEN GO TO 130
 250 LET n$="Final: ALB ="+STR$
wp+"-NEGRU="+STR$ bp
 260 LET m$="Inca un joc ?": GO
SUB 1880
 270 PAUSE 0: LET a$=INKEY$
 280 IF a$="D" OR a$="d" THEN GO
TO 110
 290 IF a$<>"N" AND a$<>"n" THEN
 GO TO 260
 300 BORDER 7: PAPER 7: INK 0: C
LS
 310 STOP
 330 LET take=0
 340 FOR v=-1 TO 1: LET dx=v
 350 FOR w=-1 TO 1: LET dy=w
 360 LET dist=0
 370 LET dist=dist+1
 380 LET ax=nx+dx*dist: LET ay=n
y+dy*dist
 390 IF ax<1 OR ax>8 OR ay<1 OR
ay>8 THEN GO TO 560
 400 IF b(ax,ay)=enemy THEN GO T
O 370
 410 IF dist=1 THEN GO TO 560
 420 IF b(nx+dx*dist,ny+dy*dist)
<>turn THEN GO TO 560
 425 LET take=take+dist-1
 430 IF moving=0 THEN GO TO 560
 440 IF take<>dist-1 THEN GO TO
520
 450 LET x=nx: LET y=ny
 460 IF x=lx THEN LET lx=x-1: IF
 lx=0 THEN LET lx=1
 470 IF x=hx THEN LET hx=x+1: IF
 hx=9 THEN LET hx=8
 480 IF y=ly THEN LET ly=y-1: IF
 ly=0 THEN LET ly=1
 490 IF y=hy THEN LET hy=y+1: IF
 hy=9 THEN LET hy=8
 500 IF turn=2 THEN GO SUB 1650:
 GO TO 510
 505 IF turn=3 THEN GO SUB 1760
 510 IF turn=3 THEN LET wp=wp+1:
 GO TO 520
 515 LET bp=bp+1
 520 FOR z=1 TO dist-1
 530 LET x=nx+dx*z: LET y=ny+dy*
z
 540 IF turn=2 THEN GO SUB 1650:
 GO TO 550
 545 IF turn=3 THEN GO SUB 1760
 550 NEXT z
 560 NEXT w
```

```
 570 NEXT v
 580 BEEP 0.01,2+take: RETURN
 600 RESTORE 690: FOR x=1 TO 4
 610 FOR y=1 TO 4
 620 READ tactic
 630 LET t(x,y)=tactic
 640 LET t(x,9-y)=tactic
 650 LET t(9-x,9-y)=tactic
 660 LET t(9-x,y)=tactic
 670 NEXT y
 680 NEXT x
 690 DATA 10,1,7,5
 700 DATA 1,0,2,2
 710 DATA 7,2,4,3
 720 DATA 5,2,3,3
 730 RETURN
 750 LET xp=0: LET yp=0: CLS : L
ET bp=0: LET wp=0
 760 LET vib=1500: LET var=1.06:
 LET n$="": GO TO 850
 770 FOR x=64 TO 192 STEP 16
 780 PLOT x,175
 790 DRAW 0,-128
 800 NEXT x
 810 FOR y=175 TO 47 STEP -16
 820 PLOT 64,y
 830 DRAW 128,0
 840 NEXT y: FOR y=0 TO 17: PRIN
T AT y,0; OVER 0;"        ": NEXT
 y: RETURN
 850 FOR x=1 TO 8
 860 FOR y=1 TO 8
 870 LET b(x,y)=1
 880 NEXT y
 890 NEXT x
 900 FOR z=4 TO 5
 910 LET x=z
 920 LET y=z: GO SUB 1650
 930 LET y=9-x: GO SUB 1760
 940 NEXT z: GO SUB 770
 950 LET bp=2: LET wp=2
 960 LET lx=3: LET ly=3: LET hx=
6: LET hy=6
 970 LET gamov=0
 980 RETURN
1000 LET m$="Citi jucatori ?": G
O SUB 1880
1010 PAUSE 0: LET p$=INKEY$
1020 IF p$<"0" OR p$>"2" THEN LE
T m$="introduceti 0,1 sau 2": GO
 SUB 1880: GO TO 1000
1030 LET n$=n$+p$
1040 IF VAL p$=0 THEN GO TO 1050
1043 IF VAL p$=1 THEN GO TO 1090
1047 IF VAL p$=2 THEN GO TO 1070
1050 LET black=1: LET white=1
1060 RETURN
1070 LET black=2: LET white=2
1080 RETURN
1090 LET m$="joci ALB sau NEGRU
?": GO SUB 1880
1100 PAUSE 0: LET c$=INKEY$
1110 IF c$="a" OR c$="A" THEN LE
T black=1: LET white=2: LET n$=n
$+"ALB": RETURN
1120 IF c$="N" OR c$="n" THEN LE
T white=1: LET black=2: LET n$=n
$+"NEGRU": RETURN
1130 GO TO 1090
1150 LET moving=0
1160 LET pts=-1000
1170 LET scraux=(hx-lx)*(hy-ly)+
1: FOR c=lx TO hx: LET nx=c
1180 FOR d=ly TO hy: LET ny=d
1190 IF b(nx,ny)<>1 THEN GO TO 1
260
1200 GO SUB 330
1210 IF take=0 THEN GO TO 1260
1220 LET scre=t(nx,ny)*20+RND*10
+take*(xp+yp-32)
1230 IF scre<pts THEN GO TO 1255
1240 LET pts=scre
1250 LET xbest=nx: LET ybest=ny
1255 LET xsc=3+37/scraux*((nx-lx
)*(hy-ly)+ny-ly+1): LET ysc=scre
/30: PLOT OVER 0;xsc,40: DRAW 0,
ysc
1260 NEXT d
1270 NEXT c
1280 LET nx=xbest: LET ny=ybest
1290 LET moving=1
1300 GO SUB 330
1310 IF bp+wp=64 THEN LET gamov=
1
1320 RETURN
1340 LET ax=xc*16+56: LET ay=183
-yc*16
1350 IF b(xc,yc)=2 THEN OVER 1:
CIRCLE ax,ay,2: OVER 0
1355 IF b(xc,yc)<>2 THEN CIRCLE
ax,ay,2
1370 OVER 1: CIRCLE ax,ay,2: OVE
R 0
1380 IF b(xc,yc)=2 THEN CIRCLE a
x,ay,2
1390 IF INKEY$="x" OR INKEY$="X"
 THEN LET xc=xc-1: IF xc<1 THEN
LET xc=1
1400 IF INKEY$="c" OR INKEY$="C"
 THEN LET xc=xc+1: IF xc>8 THEN
LET xc=8
1410 IF INKEY$="z" OR INKEY$="Z"
 THEN LET yc=yc+1: IF yc>8 THEN
LET yc=8
1420 IF INKEY$="a" OR INKEY$="A"
 THEN LET yc=yc-1: IF yc<1 THEN
LET yc=1
1430 IF INKEY$="v" OR INKEY$="V"
 AND b(xc,yc)=1 THEN GO TO 1450
1440 GO TO 1340
1450 LET nx=xc: LET ny=yc
1460 GO SUB 330
1470 IF take=0 THEN GO TO 1340
1480 LET moving=1
1490 GO SUB 330
1500 IF bp+wp=64 THEN LET gamov=
1
1510 RETURN
```

```
1530 LET moving=0: LET take=0
1540 FOR i=lx TO hx: LET nx=i
1550 FOR j=ly TO hy: LET ny=j
1560 IF b(nx,ny)<>1 THEN GO TO 1
590
1570 GO SUB 330
1580 IF take<>0 THEN LET xc=nx:
LET yc=ny: LET nx=8: LET ny=8: L
ET i=8: LET j=8
1590 NEXT j
1600 NEXT i
1610 IF take<>0 THEN LET legal=1
: RETURN
1620 IF legal=0 THEN LET gamov=1
1625 LET legal=0
1630 RETURN
1660 LET vib=INT (vib/var)
1670 IF q$="n" OR q$="N" THEN GO
 TO 1690
1680 BEEP .2,vib/100
1690 FOR n=1 TO 6
1700 CIRCLE x*16+56,183-y*16,n
1710 NEXT n
1720 LET b(x,y)=2
1730 LET wp=wp+1: LET bp=bp-1
```

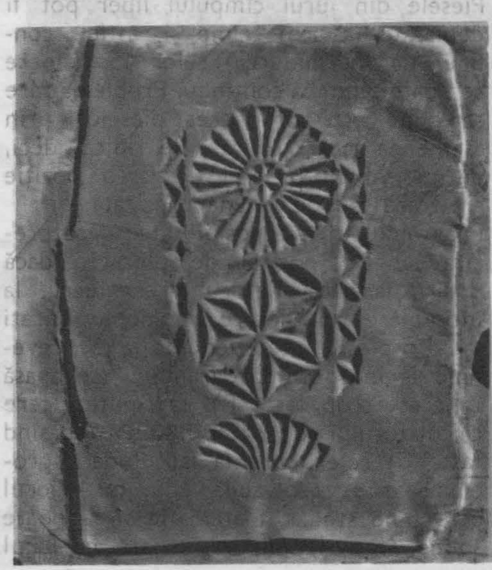
```
1740 RETURN
1770 LET vib=INT (vib*var)
1780 IF q$="n" OR q$="N" THEN GO
 TO 1800
1790 BEEP .2,vib/100
1800 LET rx=INT (x*16+48)/8: LET
 ry=22-INT (191-y*16)/8
1810 PRINT AT ry,rx;"  ";AT ry+1
,rx;"  ";
1830 CIRCLE x*16+56,183-y*16,6
1840 LET b(x,y)=3
1850 LET bp=bp+1: LET wp=wp-1
1860 RETURN
1880 PRINT AT 18,6;s$;AT 18,6;i$
;
1885 PRINT AT 19,6;s$;AT 19,6;j$
;
1890 PRINT AT 20,6;s$;AT 20,6;n$
;
1895 PRINT AT 21,5;">";s$;AT 21,
6;m$;
1900 LET i$=j$: LET j$=n$: LET n
$=m$
1910 RETURN
1920 CLS
```

JOCURI LOGICE SOLITARE

Pe lîngă jocurile de două sau mai multe persoane, există o categorie de jocuri logice care se joacă de unul singur, solitar. Numărul lor este foarte mare și majoritatea folosesc un « echipament » anumit, de natură mecanică (gen **Cubul lui Rubik**). Există însă și multe jocuri care pot fi simulate pe calculator, practicate, adică, pe ecran. Cîteva vor fi prezentate în continuare.

Caracteristica jocurilor solitare este existența unor configurații (de regulă, permutări ale unor componente) legate între ele prin « mutări » (manevre asupra jocului); problema care se pune este trecerea de la o configurație de plecare la o anumită configurație-obiectiv. De obicei, jocurile logice solitare sînt mai direct și mai puternic legate de matematică decît jocurile competitive și, într-un anume sens, însăși căutarea soluțiilor lor este o activitate de factură matematică (definirea mișcărilor permise, a efectelor lor, definirea și codificarea configurațiilor, conceperea algoritmului de rezolvare, demonstrarea că el funcționează, eficiența lui etc.). Jocul tipic de acest gen dintre cele care urmează este « 14 — 15 », al lui Sam Loyd. Am inclus în acest capitol și cele două programe referitoare la **Mastermind** (unul care « ascunde » și unul care « ghicește ») pentru că, mai ales în probleme, **Mastermind**-ul este un excelent joc solitar.

Dacă știți să rezolvați jocurile care urmează, programele de aici sînt un prilej pentru a vă dovedi măiestria. Dacă nu știți, atunci le puteți folosi pentru a experimenta diferite idei de rezolvare, ușurîndu-vă astfel găsirea unui algoritm. Iar dacă sarcina vi se pare totuși prea grea, . . .apelați la bibliografie.



JOCUL „14-15" AL LUI SAM LOYD

Acesta este unul dintre jocurile solitare care au cunoscut o popularitate deosebită, bazată atît pe calitățile ludice ca atare, cît și pe publicitatea făcută în jurul lui. Jocul a fost inventat de Sam Loyd, « cel mai mare enigmist al Americii » (M. Gardner), cu aproximativ un secol în urmă și se compune dintr-un suport pătrat, (de dimensiuni 4×4), și din 15 pătrate mici, numerotate, care se pot așeza pe acest suport. Un cîmp rămîne deci liber. Piesele din jurul cîmpului liber pot fi deplasate în acest cîmp și, astfel, configurația (permutarea) celor 15 numere se poate schimba continuu. Problema care se pune este ordonarea pieselor prin asemenea mutări (fără a le ridica, deci, de pe tablă), plecînd de la o poziție arbitrară sau de la una precizată.
Programul vă lasă posibilitatea de a preciza singuri poziția de plecare; dacă doriți, face calculatorul acest lucru, la întîmplare. În fiecare moment, puteți muta o piesă, puteți renunța la joc, revenind la începutul programului (se apasă în acest scop tasta R), sau puteți cere calculatorului o « consultație » (apăsînd tasta C) referitoare la faptul dacă problema este rezolvabilă sau nu. (Jocul este bine studiat matematic și se poate spune, într-o configurație dată, cu locul gol în colțul din dreapta-jos, dacă se poate ajunge la configurația dorită prin mutări regulamentare.) Pentru precizarea unei mutări trebuie să specificați numărul piesei care se mută, sub forma a două cifre; pentru piesele 1 — 9 se tastează un zero înainte (04 în loc de 4 etc.). Programul verifică corectitudinea mutărilor și dacă jocul este rezolvat. După încheierea cu succes a unei încercări, jocul poate fi reluat.
O modificare posibilă a programului este adăugarea unei opțiuni de « ajutor », prin activarea căreia calculatorul să preia sarcina rezolvării jocului.

Descrierea programului

10 (GOSUB 1000) — matricea **a** conține descrierea tablei de joc: pe primele două coloane sînt înregistrate coordonatele celor 16 linii pe ecran (sînt citite la liniile 1010 — 1030), iar pe coloana a treia sînt reținute piesele corespunzătoare pozițiilor respective.
30—60 — se desenează tabla de joc.
70—90 — opțiunea de completare a configurației de start.
100 — completează jucătorul.
110—150 — prima cifră.
160 — pentru că se introduc două cifre, prima este înmulțită cu 10.
170—190 — se introduce a doua cifră.
200 — piesa are numărul **nr**.
210—250 — se verifică dacă s-au introdus piese mai mari de 15 sau duble.
260 — locul gol se găsește pe linia **lg** în matricea **a**.
290 — 410 — completează calculatorul, la întîmplare.
420—500 — se introduce mutarea, din două cifre, **r$** și **t$**.
510 — piesa mutată are numărul **nr**.
540—560 — se caută piesa pe tablă (în matricea **a** — este găsită pe linia **i**).
570—580 — se verifică dacă piesa este vecină locului gol.
600—610 — se efectuează mutarea.
620—630 — se înregistrează mutarea în matricea **a** și se schimbă poziția locului gol.

640—660 — se verifică dacă s-a încheiat partida.
670—720 — încheiere cu succes, opţiune de reluare.
.730 — se evaluează configuraţia numai dacă locul gol este în dreapta-jos.

750—810 — se calculează numărul de inversiuni (variabila **inv**).
820 — se verifică paritatea numărului de inversiuni.
830—880 — mesaje şi opţiuni de continuare.

```
  10 BORDER 1: PAPER 6:: CLS : R
ESTORE : GO SUB 1000
  20 PRINT AT 2,0; INVERSE 1;" J
OCUL ""14 - 15"" AL LUI S. LOYD
"
  30 FOR i=1 TO 5
  40 PLOT 64,24+i*24: DRAW 128,0
  50 PLOT 32+i*32,48: DRAW 0,96
  60 NEXT i
  70 PRINT AT 19,1;"Cine complet
eaza (C/J) ?"
  80 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
  90 IF r$="c" THEN GO TO 290
 100 PRINT AT 19,1;"Astept sa co
mpletezi              "
 110 FOR i=1 TO 16
 120 PRINT AT a(i,1),a(i,2); FLA
SH 1;"??"
 130 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 140 IF r$<"0" OR r$>"1" THEN BE
EP 1,-6: GO TO 130
 150 PRINT AT a(i,1),a(i,2); INV
ERSE 1;r$
 160 LET nr=10*VAL r$
 170 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 180 IF r$<"0" OR r$>"9" THEN BE
EP 1,-6: GO TO 170
 190 PRINT AT a(i,1),a(i,2)+1; I
NVERSE 1;r$
 200 LET nr=nr+VAL r$
 210 IF nr>15 THEN BEEP 1,-6: GO
 TO 120
 215 IF nr=0 THEN PRINT AT a(i,1
),a(i,2);"  "
 220 IF i=1 THEN GO TO 260
 230 FOR j=1 TO i-1
 240 IF a(j,3)=nr THEN BEEP 1,-6
: GO TO 120
 250 NEXT j
 260 LET a(i,3)=nr: IF nr=0 THEN
 LET lg=i
 270 NEXT i
 280 GO TO 420
 290 PRINT AT 19,1;"Asteapta, te
 rog, sa completez"
 300 FOR i=0 TO 15
 310 LET j=INT (RND*(16-i))+1
 315 LET s=0
 320 FOR k=1 TO 16
 330 IF a(k,3)>=0 THEN GO TO 400
 340 LET s=s+1
```

```
 350 IF s<j THEN GO TO 400
 360 LET a(k,3)=i
 370 IF i=0 THEN LET lg=k: GO TO
 410
 380 IF i<=9 THEN PRINT AT a(k,1
),a(k,2); INVERSE 1;0;AT a(k,1),
a(k,2)+1;i: GO TO 410
 390>PRINT AT a(k,1),a(k,2); INV
ERSE 1;i: GO TO 410
 400 NEXT k
 410 BEEP .1,12: NEXT i
 420 LET mut=0
 425 PRINT AT 19,1;"Mut.(01-15),
rel.(R), cons.(C)"
 430 LET mut=mut+1: PRINT AT 21,
1;"Mutarea nr. ";mut;" =     "
;
 440 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 450 IF r$="r" THEN GO TO 10
 460 IF r$="c" THEN GO TO 730
 470 IF r$<"0" OR r$>"1" THEN BE
EP 1,-6: GO TO 440
 480 LET nr=10*VAL r$
 490 PAUSE 0: LET t$=INKEY$: BEE
P .1,12
 500 IF t$<"0" OR t$>"9" THEN BE
EP 1,-6: GO TO 490
 510 LET nr=nr+VAL t$
 520 IF nr>15 THEN BEEP 1,-6: GO
 TO 440
 530 PRINT AT 21,18;r$;t$
 540 FOR i=1 TO 16
 550 IF nr=a(i,3) THEN GO TO 570
 560 NEXT i
 570 IF a(i,1)=a(lg,1) AND ABS (
a(i,2)-a(lg,2))=4 THEN GO TO 600
 580 IF a(i,2)=a(lg,2) AND ABS (
a(i,1)-a(lg,1))=3 THEN GO TO 600
 590 BEEP 1,-6: PRINT AT 21,18;"
  ": GO TO 440
 600 PRINT AT a(i,1),a(i,2);"  "
 610 PRINT AT a(lg,1),a(lg,2); I
NVERSE 1;r$;t$: BEEP .05,12: BEE
P .03,22
 620 LET a(i,3)=0: LET a(lg,3)=n
r
 630 LET lg=i
 640 FOR i=1 TO 15
 650 IF a(i,3)<>i THEN GO TO 430
 660 NEXT i
 670 FOR j=1 TO 3: FOR i=1 TO 15
: BEEP .03,RND*40+i: NEXT i: NEX
T j
```

```
 680 PRINT AT 19,1;"FELICITARI -
 Ai reusit !!!        "
 690 PRINT AT 21,1;"Alt joc (d/n
) ?                   "
 700 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 710 IF r$="d" THEN GO TO 10
 720 STOP
 730 IF a(16,3)<>0 THEN PRINT AT
 19,1;"Muta locul gol in coltul
S - E": BEEP 1,-6: PAUSE 60: GO
TO 425
 740 PRINT AT 19,1;"Asteapta, te
 rog, putin          "
 745 LET inv=0: PRINT AT 21,1;"
                              "
 750 FOR i=1 TO 15
 760 IF a(i,3)=0 THEN GO TO 810
 770 FOR j=i+1 TO 16
 780 IF a(j,3)=0 THEN GO TO 800
 790 IF a(i,3)>a(j,3) THEN LET
nv=inv+1
 800 NEXT j
 810 NEXT i
 820 IF inv=2*INT (inv/2) THEN G
O TO 850
 830 PRINT AT 19,1;"Probl. impos
ibila - regret... "
 840 GO TO 690
 850 PRINT AT 19,1;"Probl. posib
ila-cont. (d/n) ? "
 860 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 870 IF r$<>"d" THEN GO TO 690
 880 GO TO 425
1000 DIM a(16,3)
1010 FOR i=1 TO 16: FOR j=1 TO 2
1020 READ a(i,j): NEXT j: NEXT i
1030 DATA 5,9,5,13,5,17,5,21,8,9
,8,13,8,17,8,21,11,9,11,13,11,17
,11,21,14,9,14,13,14,17,14,21
1040 FOR i=1 TO 16
1050 LET a(i,3)=-1
1060 NEXT i: RETURN
```

TURNUL DIN HANOI

Deşi numit astfel, jocul a fost inventat (cam cu un secol în urmă) de matematicianul francez E. Lucas. Se dau trei tije (în program sînt numerotate cu 1, 2, 3) şi un număr de discuri neegale (şase, în program) aşezate în ordine descrescătoare pe una din tije. Problema care se pune este deplasarea întregului « turn » pe o altă tijă, mutînd repetat, piesă cu piesă, cîte una la fiecare mutare, astfel încît în nici un moment o piesă să nu fie aşezată pe o piesă de dimensiune inferioară. Jocul are soluţie, dar găsirea ei nu este un lucru prea simplu.

În program, pentru indicarea unei mutări trebuie precizate numerele tijelor de plecare şi de sosire. Programul verifică corectitudinea mutărilor şi încheierea cu succes a jocului. În orice moment, este posibilă renunţarea la joc şi revenirea la poziţia de start (se apasă pentru aceasta tasta R).

După orice încercare reuşită, jocul poate fi reluat.

Două **modificări** care pot fi aduse programului sînt: a) introducerea unei opţiuni privind dimensiunea turnului (numărul de discuri) şi b) adăugarea unei opţiuni de « ajutor », la activarea căreia programul să continue singur rezolvarea jocului.



Descrierea programului

10(GOSUB 1000) — în matricea **p$** sînt desenate cele şase discuri.

20 — matricea **a** descrie starea jocului la un moment dat (ordinea discurilor pe cele trei tije).

30—50 — se completează prima linie a matricei **a**.

70—90 — se desenează tijele.

95—120 — se desenează dicurile pe tija 1.

150—2000 — se indică tija de plecare (sau opţiunea de reluare) — variabila **r**.

210—250 — se indică tija de sosire — variabila **t**.

260 — se verifică dacă pe tija **r** există piese.

270—300 — se caută piesa din vîrful tijei **r** (locul ei este identificat de variabila **lr**, iar dimensiunea piesei de variabila **pr**).

310—340 — se caută piesa din vîrful tijei **t** (locul ei

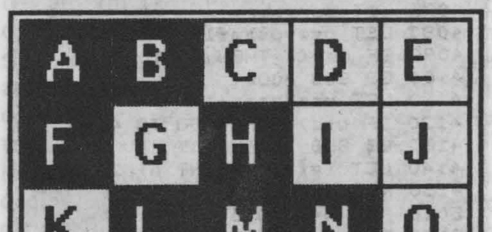
este indicat de variabila **lt**, iar piesa este identificată de variabila **pt**).
350 — se verifică dacă mutarea este corectă.
360—400 — se efectuează mişcarea.
405 — se reface tija **r**.
410 — înregistrarea mutării în matricea **a**.
420—440 — se verifică dacă jocul s-a încheiat.
458—500 — încercare reuşită şi opţiune de reluare.

```
 10 BORDER 1: INK 1: CLS : GO S
UB 1000
 20 DIM a(3,6)
 25 PRINT AT 2,6; INVERSE 1;"
TURNUL DIN HANOI  "
 30 FOR i=1 TO 6
 40 LET a(1,i)=7-i
 50 NEXT i
 60 PLOT 16,63: DRAW 220,0
 70 FOR i=1 TO 3
 80 PLOT 46+(i-1)*80,64: DRAW 0
,64: DRAW 3,0: DRAW 0,-64
 85 PRINT AT 15,5+(i-1)*10;i
 90 NEXT i
 95 OVER 1
100 FOR i=1 TO 6
110 PRINT AT 14-i,2;p$(7-i)
120 BEEP .1,i*10: NEXT i
130 OVER 0: LET mut=1
140 PRINT AT 21,1;"(Reluare = "
; INVERSE 1;"R"; INVERSE 0;")"
150 PRINT AT 19,1;"Mutarea nr.
";mut;" =      ";
160 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
170 IF r$="r" THEN GO TO 10
190 IF r$<"1" OR r$>"3" THEN BE
EP 1,-6: GO TO 160
200 PRINT AT 19,18;r$;"-";
210 PAUSE 0: LET t$=INKEY$: BEE
P .1,12
220 IF t$<"1" OR t$>"3" THEN BE
EP 1,-6: GO TO 210
230 PRINT AT 19,20;t$
240 IF r$=t$ THEN BEEP 1,-6: GO
 TO 150
250 LET r=VAL r$: LET t=VAL t$
260 IF a(r,1)=0 THEN BEEP 1,-6:
 GO TO 150
270 FOR i=1 TO 5
280 IF a(r,i+1)=0 THEN LET lr=i
: LET pr=a(r,i): GO TO 310
290 NEXT i
300 LET lr=6: LET pr=a(r,6)
310 IF a(t,1)=0 THEN LET lt=1:
GO TO 360
320 FOR i=1 TO 5
330 IF a(t,i+1)=0 THEN LET lt=i
+1: LET pt=a(t,i): GO TO 350
340 NEXT i
350 IF pr>pt THEN BEEP 1,-6: GO
 TO 170
360 FOR i=0 TO 6
370 PRINT AT 14-lr,2+10*(r-1);
INK i; OVER 1;p$(pr)
380 PRINT AT 14-lt,2+10*(t-1);
INK 7-i; OVER 1;p$(pr)
390 BEEP .1,i*6
400 NEXT i
405 PLOT 46+(r-1)*80,56+8*lr: D
RAW 0,8*(9-lr): DRAW 3,0: DRAW 0
,-8*(9-lr)
410 LET a(r,lr)=0: LET a(t,lt)=
pr
415 IF t=1 THEN LET mut=mut+1:
GO TO 150
420 FOR i=1 TO 6
430 IF a(t,i)<>7-i THEN LET mut
=mut+1: GO TO 150
440 NEXT i
450 PRINT AT 19,1;"FELICITARI -
Ai reusit !!!"
460 FOR j=1 TO 3: FOR i=1 TO 15
: BEEP .1,RND*30+i: NEXT i: NEXT
 j
470 PRINT AT 19,1;"Alt joc (d/n
) ?"
475 PRINT AT 21,1;"
"
480 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
490 IF r$="d" THEN GO TO 10
500 STOP
1000 DIM p$(6,10)
1010 LET p$(1)="   <CAPS 88>   "
1020 LET p$(2)="  <5><CAPS 885>
 "
1030 LET p$(3)="  <CAPS 8888>  "
1040 LET p$(4)=" <5><CAPS 88885>
 "
1050 LET p$(5)=" <CAPS 888888> "
1060 LET p$(6)="<5><CAPS 8888885
>"
1070 RETURN
```

Mefisto

Pe ecran se afişează un caroiaj de 5×5 pătrate, fiecare pătrat fiind identificat printr-o literă a alfabetului de la **A** la **Y**. Scopul jocului este colorarea caroiajului. Prin tastarea unei litere ce se află într-unul din pătrate, de exemplu **H**, pătratul respectiv îşi va schimba culoarea, acelaşi lucru întîmplîndu-se şi cu pătratele vecine de pe orizontală şi de pe verticală, formîndu-se o cruce. Dacă se va indica un pătrat de pe marginea caroiajului, vor fi afectate numai pătratele care ar fi format în mod normal crucea. De exemplu:

Fig.15



dacă se indică **A,** se vor colora pătratele **A**, **B**, **F** (vezi figura 15). Dacă prin indicarea colorării unei cruci va fi afectat un pătrat (sau mai multe), care a fost deja colorat, acest pătrat va reveni la culoarea iniţială. În joc un pătrat poate fi afectat sau neafectat. În afară de colorarea întregului caroiaj, se pot alege diverse alte scopuri de către jucător la începutul jocului: formarea, prin pătrate colorate, a unui table de şah, a unei figuri geometrice, colorarea întregului caroiaj cu excepţia unei pătrat sau a două pătrate etc. Colorarea întregului caroiaj (problema nu este simplă!) se poate obţine prin următoarele mutări: **A, B, F, G, M, N, O, Q, R, S, V, W, Y.** Jocul se poate relua (restarta) în orice situaţie acţionîndu-se #(SS + 3).

Descrierea programului

500—999 — sînt apelate succesiv toate subrutinele programului pînă la rezolvarea jocului sau restartarea sa.
1000—1100 — subrutina de iniţializare a tuturor variabilelor.
2000—2120 — subrutina care realizează inversarea valorii (culorii) unui pătrat (din « aprins » în « stins » şi invers).
3000—3100 — subrutina prin care se introduce şi se verifică mutarea jucătorului.
3075 — dacă jucătorul acţionează tasta corespunzătoare caracterului #(SS şi 3) jocul se va relua de la început (cu RUN).
4000—4150 — subrutină care realizează inversarea valorii pătratelor vecine cu cel indicat de jucător.
5000—5090 — subrutină care controlează rezolvarea corectă a jocului: dacă poziţia la care s-a ajuns nu este bună, se continuă jocul, aşteptîndu-se o nouă mutare; dacă poziţia la care s-a ajuns este bună, se alege între un joc nou şi abandonarea jocului.

Variabile mai importante în program

t — numărul de mutări efectuate de jucător în jocul prezent. Are valoarea iniţială **0.**
b$** — matrice bidimensională de caractere. **b$(x,y) reprezintă caracterul care se

află în celula caracter de pe linia **x** și coloana **y**. Caracterul poate fi o literă mare de la **A** la **Y**. Se rezervă inițial un spațiu de memorie pentru cele 25 de caractere (**DIM b$ (5, 5)**).

i — matrice bidimensională ale cărei elemente **i(x, y)** reprezintă valoarea pătratului din celula caracter de pe linia **x** și coloana **y**. Se rezervă inițial un spațiu de memorie pentru cele 25 de valori (**DIM i (5×5)**). În funcție de valoarea pătratului se stabilește culoarea lui astfel:

dacă **i = 1** pătratul va fi «aprins»;
dacă **i = 0** pătratul va fi «stins».

sz — codul ASCII al caracterului care urmează să fie scris pe tabla de joc în timpul desenării acesteia. Inițial valoarea variabilei este 65 (codul ASCII corespunzător literei **A**) și apoi se incrementează pînă la 90 (codul ASCII corespunzător literei **Z**).

sor, osz — reprezintă coordonatele caracterului care este examinat. De exemplu, **b$ (sor, osz)** este caracterul care se află pe linia **sor** și pe coloana **osz.**

n$** — reprezintă caracterul introdus de jucător; după introducerea mutării se caută coordonatele unde se află **n$ și se inversează apoi valoarea pătratului corespunzător și a pătratelor vecine.

sorm, oszm — reprezintă coordonatele (linie, coloană) cu care se lucrează cînd se modifică valoarea pătratului în care se găsește caracterul aflat la **(sor, osz)**. Se face succesiv controlul posibilității de a modifica și valoarea pătratelor vecine cu cel ales.

```
 510 GO SUB 1000
 511 LET t=0
 520 GO SUB 3000
 530 GO SUB 2000
 540 GO SUB 4000
 550 GO SUB 5000
 999 GO TO 520
1001 CLS : DIM b$(5,5): DIM i(5,
5): LET sz=65: PAPER 6: INK 1
1002 FOR d=1 TO 22: PRINT "
                         ": NEX
T d
1010 FOR q=1 TO 5
1020 FOR w=1 TO 5
1030 LET b$(q,w)=CHR$ sz
1040 LET sz=sz+1
1050 LET i(q,w)=1
1060 LET sor=q
1070 LET osz=w
1080 GO SUB 2000
1090 NEXT w
1100 NEXT q
1110 RETURN
2010 IF i(sor,osz)=1 THEN GO TO
2100
2020 LET i(sor,osz)=1
2030 PRINT AT (sor*2)+5,(osz*2)+
8; INVERSE 1;b$(sor,osz); INVERS
E 0
2040 RETURN
2100 LET i(sor,osz)=0
2110 PRINT AT sor*2+5,osz*2+8;b$
(sor,osz)
2120 RETURN
3010 LET n=0
3020 LET n$=INKEY$: IF n$="" THE
N GO TO 3020
```

```
3022 LET n$=CHR$ (CODE n$-32)
3030 FOR q=1 TO 5
3040 FOR w=1 TO 5
3050 IF n$=b$(q,w) THEN GO TO 30
90
3060 NEXT w
3070 NEXT q
3075 IF CODE n$+32=35 THEN RUN
3080 GO TO 3020
3090 LET sor=q: LET osz=w
3100 RETURN
4010 LET sorm=sor: LET oszm=osz
4020 LET sor=sorm-1
4030 IF sor<1 THEN GO TO 4050
4040 GO SUB 2000
4050 LET sor=sorm+1
4060 IF sor>5 THEN GO TO 4075
4070 GO SUB 2000
4075 LET sor=sorm
4080 LET osz=oszm-1
4090 IF osz<1 THEN GO TO 4110
4100 GO SUB 2000
4110 LET osz=oszm+1
4120 IF osz>5 THEN GO TO 4140
4130 GO SUB 2000
4140 LET t=t+1: PRINT AT 20,28;t
4150 RETURN
5010 FOR q=1 TO 5
5020 FOR w=1 TO 5
5030 IF i(q,w)=0 THEN RETURN
5040 NEXT w
5050 NEXT q
5060 CLS : PRINT AT 10,5;"FELICI
TARI"
5070 INPUT "ALT JOC ? ";v$
5080 IF v$="n" THEN STOP
5090 RUN
```

DAME

(VARIANTĂ)

Este o variantă chinezească a jocului tradițional de dame, bazată pe mutarea pieselor pe diagonală.
Pe ecran apare o tablă de joc care are două părți cu poziții (căsuțe) de culoare închisă aranjate pe diagonală. Căsuțele alăturate (de culoare deschisă) au numai rolul de a marca cu un număr căsuța de deasupra. Sînt 8 căsuțe cu semne de un anumit tip (>) într-o parte și 8 căsuțe cu semne de alt tip (<) în partea opusă. În centru există o singură căsuță goală, care are rol de punte de trecere pentru semne. Căsuțele sînt numerotate de la stînga la dreapta în ordine, fiind în total 17 căsuțe.
Scopul jocului este de a se aduce toate semnele din partea stîngă în partea dreaptă și invers. Mutările se fac indicîndu-se numărul căsuței în care se află semnul care se dorește a fi mutat. Un semn poate fi mutat numai în direcția pe care el o indică (deci spre dreapta pentru semnele «>» și spre stînga pentru semnele «<»), în căsuța imediat următoare de pe diagonală, cu condiția ca aceasta să fie căsuță goală (liberă). De asemenea, un semn poate sări peste o căsuță (și numai peste una) cu un semn, pe diagonală, în căsuța goală (mutarea specifică jocului dame). În partea superioară a ecranului se indică scorul, care arată numărul de mutări de piese (semne) care s-a reușit a se efectua, precum și scorul maxim realizat (recordul).
În partea de jos a ecranului se afișează mesajul de introducere a mutării.

Descrierea programului

20 — alocarea spațiului de memorie pentru variabile.
Variabila **A$** va ține minte configurația semnelor din căsuțe la un moment dat, **R** și **B** sînt variabile de control asupra tablei de joc și a pieselor.
30 — inițializare variabile: **S4** reprezintă recordul (numărul maxim de mutări realizate într-o sesiune de jocuri; inițial este **0**); **Q$** — variabilă șir de caractere care ține minte configurația finală de semne la care trebuie să se ajungă. Această configurație va avea primele 8 căsuțe ocupate cu semnele «<», ultimele 8 cu semnele «>», căsuța 9 fiind goală.
60 — afișarea mesajului din partea superioară a ecranului.
70 — inițializarea variabilei **S3** care reprezintă numărul de mutări efectuate.
80 — inițializarea variabilei **A$**; stabilirea semnelor pentru fiecare căsuță în momentul începerii jocului.
90—260 — date pentru variabilele **R** și **B**
280—300 — ințializarea variabilei **R** cu datele citite.
310—330 — inițializarea variabilei **B** cu datele citite.
340 (GOSUB 700) — apelarea subrutinei de desenare a unei părți din tabla de joc. Fiecare căsuță este un pătrat de 3×3 celule caracter. Subrutina desenează toate căsuțele în care se pot muta semnele, adică pentru fiecare căsuță înnegrește toate cele 9 celule caracter.
370 — date pentru desenarea tablei de joc (liniile și coloanele).
380—410 — desenarea semnelor pe tabla de joc;

dacă există un semn într-o căsuță, acesta se va desena în celula caracter din mijlocul căsuței.

420 — se afișează numărul de mutări efectuate în partea de sus a ecranului.

430 — se inițializează variabila **T$**, care va memora configurația semnelor de pe tabla de joc după fiecare mutare.

440—450 — se calculează variabila **T$** în funcție de valoarea variabilei **A$**.

460 — se compară variabilele **T$** și **Q$**, care conțin configurația actuală a semnelor și respectiv cea la care se dorește să se ajungă; dacă valorile celor două variabile sînt diferite, se trece la efectuarea următoarei mutări.

500 — salt spre rutina de calcul a scorului și recordului.

510 — introducere mutare: variabila **G.**

510—520 — respingerea mutărior nelegale.

530 — dacă la mutare s-a indicat o căsuță goală, se va trece la rutina de semnalare a unei mutări greșite.

540 — mutare corectă; se calculează scorul și se fac calculele pentru reconfigurarea tablei de joc.

630 — numărul de mutări efectuate crește cu o unitate.

640 — redesenarea tablei de joc cu noua configurație a semnelor.

650 — dacă recordul a fost depășit, numărul de mutări la care s-a ajuns va deveni noul record.

660 — afișarea recordului.

670 — alt joc?

690 — sfîrșit.

```
  20 DIM A$(17): DIM R(17,4): DI
M B(17,4)
  30 LET S4=0: LET Q$="<<<<<<<<
>>>>>>>>"
  60 PRINT AT 3,9;"S C O R :  ";
AT 4,11; INK 0;"RESTART"
  70 LET S3=0
  80 FOR N=1 TO 8: LET A$(N)=">"
: LET A$(N+9)="<": NEXT N: LET A
$(9)=" "
  90 DATA 2,3,4,6,4,5,8,0
 100 DATA 5,6,7,0,7,9,0,0
 110 DATA 7,8,0,0,8,9,0,0
 120 DATA 9,11,0,0,9,10,0,0
 130 DATA 10,11,12,14,12,13,16,0
 140 DATA 13,14,15,0,15,17,0,0
 150 DATA 15,16,0,0,16,17,0,0
 160 DATA 17,0,0,0,17,0,0,0
 170 DATA 0,0,0,0
 180 DATA 0,0,0,0,1,0,0,0
 190 DATA 1,0,0,0,2,1,0,0
 200 DATA 3,2,0,0,3,1,0,0
 210 DATA 5,4,3,0,6,5,2,0
 220 DATA 8,7,6,4,9,8,0,0
 230 DATA 9,7,0,0,10,9,0,0
 240 DATA 11,10,0,0,11,9,0,0
 250 DATA 13,12,11,0,14,13,10,0
 260 DATA 16,15,14,12
 270 RESTORE
 280 FOR N=1 TO 17: FOR M=1 TO 4
 290 READ R(N,M)
 300 NEXT M: NEXT N
 310 FOR N=1 TO 17: FOR M=1 TO 4
 320 READ B(N,M)
 330 NEXT M: NEXT N
 340 GO SUB 700
 360 RESTORE 370
 370 DATA 12,3,9,6,15,6,6,9,12,9
,18,9,9,12,15,12,12,15,9,18,15,1
8,6,21,12,21,18,21,9,24,15,24,12
,27
 380 FOR L=1 TO 17: READ M: READ
 N
 390 PRINT AT M,N;A$(L);"<CAPS 8
>";
 400 IF A$(L)=" " THEN PRINT AT
M,N;"<CAPS 8>";
 410 NEXT L
 420 PRINT AT 3,21;S3: BEEP .1,1
0
 430 LET T$=""
 440 FOR L=1 TO 17
 450 LET T$=T$+A$(L): NEXT L
 460 IF T$<>Q$ THEN GO TO 510
 470 FOR L=1/2 TO 15 STEP 3/4
 480 PRINT AT 1,8; FLASH 1;"  B
R A V O  ! ";: BEEP 3/100,2*L

 490 PRINT AT 1,8;"
 ";: BEEP 1/100,L*4: NEXT L
 500 PRINT : GO TO 650
 510 INPUT AT 0,8;" MUTAREA TA :
";G: IF G>17 THEN GO TO 780
 520 IF G<1 THEN GO TO 650
 530 IF A$(G)=" " THEN GO TO 780
 540 IF A$(G)="<" THEN GO TO 590
 550 FOR N=1 TO 4: LET S=R(G,N)
 560 IF S=0 THEN GO TO 780
 570 IF A$(S)=" " THEN LET A$(S)
=">": LET A$(G)=" ": GO TO 630
 580 NEXT N: GO TO 780
 590 FOR N=1 TO 4: LET S=B(G,N)
 600 IF S=0 THEN GO TO 780
 610 IF A$(S)=" " THEN LET A$(S)
="<": LET A$(G)=" ": GO TO 630
 620 NEXT N: GO TO 780
 630 LET S3=S3+1
```

```
640 GO TO 350
650 IF S4<S3 THEN LET S4=S3
660 PRINT AT 2,5;" SCORUL MAXIM
ESTE ";S4;" "
670 PRINT AT 21,8; INK 0;" INCA
UN JOC ?": INK 7
680 PAUSE 0: IF INKEY$="D" OR I
NKEY$="d" THEN PRINT AT 21,8;"
               ": GO TO 50
690 CLS : STOP
700 FOR L=1 TO 17
710 READ M: READ N
720 PRINT AT M-1,N-1;"<CAPS 888
```



Acesta este unul din cele mai interesante jocuri logico-matematice, cu o mare răspîndire cu aproximativ un deceniu și jumătate în urmă. Varianta uzuală folosește un suport de joc, ciupercuțe colorate (în șase culori), cuișoare albe și negre pentru evaluarea « mutărilor » etc. (Jocul este descris în amănunțime în **Cartea jocurilor,** RECOOP, 1988, pag. 32—34.)
Programul de aici vă propune să jucați **Mastermind** cu numere, de fapt, să ghiciți combinația pe care el o ascunde. Sînt ascunse patru cifre, între 1 și 6. Programul vă întreabă la început dacă doriți să jucați « Cu repetări sau fără repetări ? » și are grijă să aleagă o combinație de genul dorit de dumneavoastră.

```
730 PRINT AT M,N-1;"<CAPS 888>"
740 PRINT AT M+1,N-1;"<CAPS 888
>"
750 PRINT AT M+2,N-1;L
760 NEXT L
770 RETURN
780 FOR L=0 TO 7
790 PRINT #1;TAB 8;" G R E S I
T ! ": BEEP .1,-10
800 INPUT :: BEEP .15,0
810 NEXT L
820 GO TO 510
1225 GO TO 1200
9999 SAVE "dame" LINE 0: VERIFY
"dame"
```

Trebuie, apoi, să propuneți combinații de cîte patru cifre, în încercarea de a le deduce pe cele ascunse. Pentru fiecare cifră ghicită complet, inclusiv locul deci, programul scrie pe ecran un asterisc (cuișor alb, în varianta obișnuită); pentru o cifră ghicită, dar nu la locul potrivit, se scrie pe ecran un semn plus (cuișor negru); dacă nu s-a făcut nici o identificare de cifre, se tipărește o liniuță. Se continuă astfel pînă la completa identificare a cifrelor ascunse.

Descrierea programului

5 — în vectorul **a** este păstrată combinația ascunsă de calculator, iar în vectorul **b** se va afla propunerea jucătorului.
10—50 — instrucțiunile de joc.
55—65 — opțiunea « cu repetări »/« fără repetări ».
70—75 — alegerea combinației (de două ori, pentru o « amestecare » mai bună a numerelor), în cazul cu repetări.
80 — **I** este numărul curent de încercări.
90—135 — se introduce propunerea jucătorului.
140 — **s** este numărul de cifre complet identificate (locul și valoarea, cuișoare albe, în jocul uzual), iar **p** este numărul de cifre parțial identificate (la alt loc, cuișoare negre).
145—170 — numărarea pozițiilor complet identificate.
180—230 — numărarea cifrelor identificate la altă poziție.

240—285 — se prezintă evaluarea (steluțe și semne plus).
290 — nu s-a identificat nimic.
300 — identificarea completă a configurației ascunse.
303—307 — refacerea vectorului **a** (la calcularea lui **s**, pozițiile numărate au fost alterate, pentru a nu fi numărate și la calcularea lui **p**).
310—318 — mesaje de încheiere.
320—340 — opțiune de reluare.
400—500 — alegerea combinației ascunse în cazul fără repetări.

```
  5 BORDER 4: PAPER 6: CLS : DI
M a(4): DIM b(4)
 10 PRINT AT 0,6;"M A S T E R M
 I N D"
 20 PRINT AT 2,2;"Se deduc 4 ci
fre dintre 1,2,3,"
 25 PRINT "  4,5,6 (cu sau fara
 repetari)."
 30 PRINT "La incercari se rasp
unde cu:"
 40 PRINT "* = cifra ghicita la
 loc corect"
 45 PRINT "+ = cifra ghicita la
 alt loc"
 50 PRINT "- = nici o cifra ghi
cita"
 55 PRINT : PRINT "Cu repetari(
n/d) ? ";: PAUSE 0: LET r$=INKEY
$: BEEP .1,12
 60 IF r$="n" THEN GO TO 400
 65 PRINT "(cu repetari)": PRIN
T
 70 FOR i=1 TO 4: LET a(i)=INT
(RND*6)+1: NEXT i
 75 FOR i=1 TO 4: LET a(i)=INT
(RND*6)+1: NEXT i
 80 LET l=0
 85 LET l=l+1
 90 PRINT "incerc. nr ";l;": ";
: BEEP .1,22
 100 FOR i=1 TO 4
 110 PAUSE 0: LET r$=INKEY$
 120 IF r$>="1" AND r$<="6" THEN
 LET b(i)=VAL r$: GO TO 130
 125 BEEP 1,-6: GO TO 110
 130 BEEP .03,12: BEEP .03,22: P
RINT b(i);" ";
 135 NEXT i
 140 LET s=0: LET p=0
 145 FOR i=1 TO 4
 150 IF a(i)<>b(i) THEN GO TO 17
0
 160 LET s=s+1: LET a(i)=a(i)-8
 170 NEXT i
 180 FOR i=1 TO 4
 185 LET k=0
 190 FOR j=1 TO 4
 200 IF a(i)=b(j) THEN LET k=1
 210 NEXT j
 220 IF k=1 THEN LET p=p+1
 230 NEXT i
 240 IF s=0 THEN GO TO 270
 250 FOR i=1 TO s
 260 PRINT "*"; " ";: BEEP .1,10
 265 NEXT i
 270 IF p=0 THEN GO TO 290
 275 FOR i=1 TO p
 280 PRINT "+";" ";: BEEP .1,10
 285 NEXT i
 290 IF s+p=0 THEN PRINT "-": BE
EP .2,5: GO TO 300
 295 PRINT " "
 300 IF s=4 THEN GO TO 310
 303 FOR i=1 TO 4
 305 IF a(i)<0 THEN LET a(i)=a(i
)+8
 307 NEXT i: GO TO 85
 310 IF l<=4 THEN PRINT : PRINT
"Formidabil !!! Ai ghicit dupa":
 PRINT "numai ";l;" incercari !"
: GO TO 320
 312 IF l<=7 THEN PRINT : PRINT
"Felicitari: ai ghicit dupa ";l;
"    incercari": GO TO 320
 314 IF l<=10 THEN PRINT : PRINT
 "Bine: ai ghicit dupa ";l;" inc
ercari": GO TO 320
 316 IF l<=15 THEN PRINT : PRINT
 "Slab: ai ghicit abia din ";l;"
     incercari; mai invata !": G
O TO 320
 318 PRINT " Mizerabil ! Ti-au t
rebuit ";l;"   incercari..."
 320 PRINT "Alt joc (d/n) ?"
 325 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 330 IF r$="d" THEN CLS : GO TO
10
 340 STOP
 400 PRINT "(fara)": PRINT : LET
a(1)=INT (RND*6)+1
 410 LET x=INT (RND*6)+1
 420 IF a(1)=x THEN GO TO 410
 430 LET a(2)=x
 440 LET x=INT (RND*6)+1
 450 IF a(1)=x OR a(2)=x THEN GO
 TO 440
 460 LET a(3)=x
 470 LET x=INT (RND*6)+1
 480 IF a(1)=x OR a(2)=x OR a(3)
=x THEN GO TO 470
 490 LET a(4)=x
 500 GO TO 80
```

masteract

Programul joacă rolul partenerului activ la **Mastermind,** adică deduce o combinație de patru cifre, între 1 și 6, « ascunsă » de utilizator. La prima vedere, sarcina este exagerat de dificilă, fiind greu de imaginat un program care să efectueze lanțul complex de raționamente pe care îl cere jocul. Dificultatea poate fi însă ocolită apelînd la puterea « brută » de calcul. Ideea este următoarea: generăm la început toate combinațiile posibile, apoi, după fiecare propunere a programului, reținem din ele numai pe cele care conduc la aceeași evaluare cu cea dată de jucător ultimei propuneri a programului, Mulțimea combinațiilor se diminuează astfel vertiginos. Inițial, avem:
$A_6^4 = 6\times5\times4\times3 = 360$
de combinații în cazul fără repetiție și
$6^4 = 1296$
în cazul cu repetiție. Programul acceptă numai combinații fără repetiție și folosind strategia schițată mai sus, deduce surprinzător de repede orice combinație: în general, 4—5 încercări sînt de ajuns și în numai cîteva situații are nevoie de 6 încercări. După fiecare propunere, programul cere evaluarea ei, sub forma « număr de cuișoare albe », respectiv, « număr de cuișoare negre ». Pe ecran apar, de fapt, întrebările « A = ? », « N = ? » (dacă este cazul, se tastează zero.). Dacă evaluările sînt eronate, programul va sesiza acest lucru imediat sau la o încercare ulterioară (va constata că mulțimea combinațiilor acceptabile devine vidă). După deducerea unei combinații, jocul poate fi reluat (« Alt joc (d/n) ? »). În colțul din dreapta-sus al ecranului apare permanent un număr (inițial el este 360); el reprezintă numărul combinațiilor reținute la pasul respectiv. Urmărindu-l, ne putem face o idee despre viteza de descreștere a mulțimii combinațiilor din care trebuie aleasă soluția.

Modificări importante:

— Considerarea și a cazului cu repetări. Modificările în program sînt simple: generarea la început a tuturor celor 1296 de combinații (eliminarea liniilor 40—170) și calcularea adecvată a parametrilor **al** și **ne** (liniile 510—530, respectiv, 550—590). Timpul de calcul va crește, însă, de aproximativ trei ori. Pentru a obține un răspuns mai rapid, putem aplica următoarea soluție: să nu mai generăm toate combinațiile de la început, ci pe parcursul derulării jocului, în fiecare moment pe prima care urmează după ultima considerată și care are aceeași evaluare cu toate propunerile de pînă atunci ale calculatorului. În cele din urmă, se va ajunge și la combinația corectă.

— Combinarea programelor **Mastermind** și **Masteract** și realizarea unui program de competiție (o mutare jucătorul, una programul, pînă ce unul din cei doi deduce combinația ascunsă de adversar).

Descrierea programului

20 — matricea **a** conține cele 360 de aranjamente

posibile a patru cifre dintre 1, 2, 3, 4, 5, 6, iar vectorul **p** conține propunerea curentă a calculatorului.

30—170 — se completează matricea **a** (variabilele **i, j, k, l** merg de la 1 la 6 și, atunci cînd sînt toate distincte, se completează o linie în matrice — linia 130 din program).

180—200 — se așteaptă începerea jocului.

210 — vectorul **c** va indica combinațiile care conduc la aceeași evaluare cu aceea dată de utilizator ultimei propuneri a programului, implicit va indica acele combinații neeliminate încă; variabila **inc** indică numărul de încercări efectuate de program.

230—250 — se verifică dacă mai sînt combinații de examinat.

260—300 — mesaj de eroare și opțiune de reluare a jocului.

310—330 — se propune prima dintre combinațiile neeliminate încă.

340—390 — se cere evaluarea combinației curente; **alb** = numărul «cuișoarelor albe», **negru** = numărul «cuișoarelor negre».

400—440 — există posibilitatea de corectare a unei erori în evaluarea curentă, caz în care se revine la cererea unei noi evaluări.

450 — deducerea este corectă.

470 — eroare în evaluare.

475—620 — se identifică acele combinații, din cele neeliminate încă (avînd adică valoarea din vectorul **c** corespunzătoare pasului, **c(i)** = **inc** — linia 490), care au aceeași evaluare cu cea specificată de jucător pentru ultima propunere; «cuișoarele» albe și negre — variabilele **al** și **ne** de la linia 500 — sînt numărate la liniile 510 — 530, respectiv, 550—590 și comparate cu **alb** și **negru**, la liniile 540, respectiv, 600; combinațiile reținute sînt marcate la linia 610.

640 — se trece la o nouă încercare.

650—680 — mesaj de încheiere a deducerii, sonorizare și revenire (la opțiunea de reluare).

```
  10 BORDER 1: PAPER 6: CLS
  20 DIM a(360,4): DIM p(4)
  30 PRINT AT 6,2;"Asteapta, te
rog, putin."
  40 LET nr=0
  50 FOR i=1 TO 6
  60 FOR j=1 TO 6
  70 IF j=i THEN GO TO 160
  80 FOR k=1 TO 6
  90 IF k=i OR k=j THEN GO TO 15
0
 100 FOR l=1 TO 6
 110 IF l=i OR l=j OR l=k THEN G
O TO 140
 120 LET nr=nr+1
 130 LET a(nr,1)=i: LET a(nr,2)=
j: LET a(nr,3)=k: LET a(nr,4)=l
 140 NEXT l
 150 NEXT k
 160 NEXT j
 170 NEXT i
 180 PRINT AT 6,2;"OK - Incepem"
 185 PRINT AT 8,0;"Alege combina
tia (fara repetari)"
 190 PRINT "Cind esti gata apasa
 o tasta"
 200 PAUSE 0: BEEP .1,12: CLS :
PRINT AT 1,29;360
 210 DIM c(360): LET inc=0: PRIN
T
 220 PRINT AT 4+inc,0;"Incerc ";
inc+1;":";
```

```
 230 FOR i=1 TO 360
 240 IF c(i)=inc THEN GO TO 310
 250 NEXT i
 260 PRINT : PRINT "Eroare !!":
BEEP 1,-6
 270 PRINT "Alt joc (d/n) ?"
 280 PAUSE 0: IF INKEY$<>"d" THE
N STOP
 300 CLS : GO TO 180
 310 FOR j=1 TO 4
 320 LET p(j)=a(i,j): PRINT " ";
p(j);: BEEP .1*j,10*j
 330 NEXT j
 340 PRINT "  "; INVERSE 1;"=";
INVERSE 0;" A = "; FLASH 1;"?"
 350 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 360 IF r$>="0" AND r$<="4" THEN
 PRINT AT 4+inc,25;r$;" N = "; F
LASH 1;"?": LET alb=VAL r$: GO T
O 380
 370 BEEP 1,-6: GO TO 350
 380 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 390 IF r$<"0" OR r$>"4" THEN BE
EP 1,-6: GO TO 380
 400 PRINT AT 4+inc,31;r$: LET n
egr=VAL r$: PRINT " Corect (d/n)
?"
 410 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 420 IF r$="d" THEN PRINT AT 5+i
```

```
nc,1;"Asteapta         ": GO TO 45
0
 430 PRINT AT 4+inc,25;"        "
: PRINT "              "
 440 PRINT AT 4+inc,25; FLASH 1;
"?": GO TO 350
 450 IF alb=4 AND negr=0 THEN GO
 TO 650
 470 IF alb+negr>4 THEN GO TO 26
0
 475 LET nr=0: PRINT AT 1,29;"0
"
 480 FOR i=1 TO 360
 490 IF c(i)<inc THEN GO TO 620
 500 LET al=0: LET ne=0
 510 FOR j=1 TO 4
 520 IF a(i,j)=p(j) THEN LET al=
l+1
 530 NEXT j
 540 IF alb<>al THEN GO TO 620
```

```
 550 FOR j=1 TO 4
 560 FOR k=1 TO 4
 565 IF k=j THEN GO TO 580
 570 IF a(i,j)=p(k) THEN LET ne=
ne+1: GO TO 590
 580 NEXT k
 590 NEXT j
 600 IF negr<>ne THEN GO TO 620
 610 LET c(i)=c(i)+1: LET nr=nr+
1: PRINT AT 1,29;nr
 620 NEXT i
 640 LET inc=inc+1: GO TO 220
 650 PRINT : PRINT "Am reusit !!
!"
 660 FOR i=1 TO 5: FOR j=7 TO 1
STEP -1
 670 BORDER j: BEEP .02*i,i+2*j+
10
 680 NEXT j: NEXT i: GO TO 270
```

PĂTRATUL MAGIC AL LUI TRAIAN PREDAN

Printre **modificările** care se pot aduce programului, sînt introducerea posibilității de a alege dimensiunile tablei (jocul se desfășoară la fel pe orice tablă pătrată), eventual, adăugarea opțiunii de « ajutor », la activarea căreia calculatorul să preia sarcina rezolvării.

Jocul a fost propus cu cîțiva ani în urmă de T.Predan, elev pe atunci, în cadrul unui concurs de jocuri logice organizat de revista **« Știință și tehnică »** în colaborare cu **Recoop.** Pe un suport ca cel din figură, sînt așezate 16 pătrate numerotate de la 1 la 16 (în program se scrie 01, 02, . . .). Inițial, piesele sînt așezate fie la întîmplare, fie într-o ordine anumită. Problema care se pune este ordonarea pieselor prin mișcări de tipul următor: orice șir de patru piese, așezate orizontal sau vertical, poate fi deplasat spre capetele sale, cu un pas sau, dacă se poate, cu doi (nu pot fi, deci, deplasate șiruri de mai puțin de patru piese și nu se pot depăși marginile tablei).

Ca și în cazul altor jocuri de permutare, numai configurațiile pare (cu un număr par de inversiuni) pot fi ordonate prin mutări regulamentare.

În program, mutările sînt indicate prin precizarea unei litere, conform figurii: **a** — **d** indică deplasarea liniei corespunzătoare spre stînga cu un pas, **e** —**h** indică deplasarea spre dreapta și așa mai departe. Configurația de plecare poate fi specificată de calculator sau de jucător (programul verifică dacă se introduc « piese » distincte). În fiecare moment, se poate, fie efectua o mutare, fie se poate cere reluarea jocului (tasta X) sau o « consultație » referitoare la faptul dacă problema este rezolvabilă sau nu (tasta Z). La această opțiune se răspunde numai dacă piesele sînt așezate în pătratul central.

După fiecare mutare, programul verifică dacă jocul este încheiat; în caz afirmativ se întreabă dacă se dorește reluarea lui.

Descrierea programului

20 — în matricea **a$** este memorată configurația curentă a tablei, **s$** și **d$** conțin inscripțiile care se scriu în partea stîngă și dreaptă a ecranului, iar **c$** conține șirurile « 01 », « 02 », . . ., « 16 ».

30—80 — se scrie numele jocului pe coloanele din stînga și din dreapta ecranului.

90—160 — se desenează tabla de joc.

170—180 — se încarcă vectorul **c$** (folosit pentru completarea de către calculator a configurației inițiale).

190—200 — se încarcă cu spații matricea **a$.**

210—220 — se încarcă vectorul **l$** (folosit pentru marcarea tablei)

230—280 — se marchează tabla.

310—330 — opțiunea de completare.

340—540 — jucătorul completează configurația de start (fiecare cîmp va conține două simboluri, **r$** și **t$,** introduse la liniile 380 respectiv, 410); dacă valoarea nu a mai fost introdusă (se verifică aces-

lucru la liniile 480—500), ea este acceptată, tipărită pe ecran și introdusă în matricea **a$** (linia 510).
550—680 — calculatorul completează configurația de plecare, alegînd la întîmplare (linia 570) cîte o valoare pentru fiecare cîmp.
700—715 — se introduce o mutare.
720 — opțiunea de reluare.
730 — opțiunea «consultație».
740—760 — mutarea este identificată (ca poziție în vectorul **l$**).
780—810 — se examinează mutarea (linie sau coloană ? în ce direcție?).
820—910 — se efectuează o mutare verticală, în jos, pe coloana **col**.
920—1010 — se efectuează o mutare verticală, în sus, pe coloana **col**.
1020—1110 — se efectuează o mutare orizontală, spre dreapta, pe linia **lin**.
1120—1210 — se efectuează o mutare orizontală, spre stînga, pe linia **lin**.
De fiecare dată, se verifică dacă mutarea este regulamentară (dacă există patru piese pe linia/coloana respectivă) și dacă ea este posibilă (dacă există cîmpuri libere în direcția specificată).
1220—1240 — se verifi ă încheierea jocului.
1250—1320 — mesaje de încheiere, opțiune de reluare.
1330—1380 — se mută «piesele» din matricea **a$** în vectorul **v$**, prilej cu care se verifică dacă toate se găsesc în pătratul central.
1390—1420 — se numără inversiunile (în vectorul **v$**; se rețin în variabila **inv**)
1430—1490 — mesaje, opțiuni de continuare.

```
  10 RESTORE : BORDER 1: PAPER 6
: CLS
  20 DIM a$(6,6,2): DIM s$(22):
DIM d$(22): DIM c$(16,2)
  30 LET s$="PATRATUL MAGIC AL
  "
  40 LET d$="LUI TRAIAN PREDAN
   "
  50 FOR i=0 TO 21
  60 PRINT AT i,0; INVERSE 1;s$(
i+1)
  70 PRINT AT i,31; INVERSE 1;d$
(i+1)
  80 NEXT i
  90 FOR i=1 TO 5
 100 PLOT 32,24+24*i: DRAW 191,0
 110 PLOT 32+32*i,24: DRAW 0,143
 120 NEXT i
 130 PLOT 32,48: DRAW 0,96
 140 PLOT 223,48: DRAW 0,96
 150 PLOT 64,24: DRAW 128,0
 160 PLOT 64,167: DRAW 128,0
 170 FOR i=1 TO 16: READ c$(i):
NEXT i
 180 DATA "01","02","03","04","0
5","06","07","08","09","10","11"
,"12","13","14","15","16"
 190 FOR i=1 TO 6: FOR j=1 TO 6
 200 LET a$(i,j)="  ": NEXT j: N
EXT i
 210 DIM l$(16)
 220 LET l$="abcdefghijklmnop"
 230 FOR i=1 TO 4
 240 PRINT AT 2+3*i,3;l$(i)
 250 PRINT AT 2+3*i,28;l$(4+i)
 260 PRINT AT 0,6+4*i;l$(8+i)
 270 PRINT AT 19,6+4*i;l$(12+i)
```

```
 280 NEXT i
 310 PRINT AT 21,1;"Cine complet
eaza (C/J) ?"
 320 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 330 IF r$="c" THEN GO TO 550
 340 PRINT AT 21,1;"Astept sa co
mpletezi        "
 350 FOR i=1 TO 4
 360 FOR j=1 TO 4
 370 PRINT AT 2+3*i,5+4*j; FLASH
 1;"??"
 380 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 390 IF r$<"0" OR r$>"1" THEN BE
EP 1,-6: GO TO 380
 400 PRINT AT 2+3*i,5+4*j; INVER
SE 1;r$
 410 PAUSE 0: LET t$=INKEY$: BEE
P .1,12
 420 IF t$<"0" OR t$>"9" THEN BE
EP 1,-6: GO TO 410
 460 IF r$="1" AND t$>"6" THEN B
EEP.1,-6: GO TO 410
 465 IF r$="0" AND t$="0" THEN B
EEP 1,-6: GO TO 370
 470 PRINT AT 2+3*i,6+4*j; INVER
SE 1;t$
 480 FOR k=1 TO i: FOR l=1 TO j
 490 IF a$(k+1,l+1,1)=r$ AND a$(
k+1,l+i,2)=t$ THEN BEEP 1,-6: GO
 TO 370
 500 NEXT l: NEXT k
 510 LET a$(i+1,j+1,1)=r$: LET a
$(i+1,j+1,2)=t$
 520 NEXT j
 530 NEXT i
```

```
 540 GO TO 700
 550 PRINT AT 21,1;"Asteapta, te
 rog, putin        "
 560 FOR i=1 TO 16
 570 LET j=INT (RND*(17-i))+1
 580 LET s=0
 590 FOR k=2 TO 5: FOR l=2 TO 5
 600 IF a$(k,l)<>"  " THEN GO TO
 660
 620 LET s=s+1
 630 IF s<j THEN GO TO 660
 640 LET a$(k,l)=c$(i)
 650 PRINT AT 3*k-1,1+4*l; INVER
SE 1;c$(i): BEEP .02,12: BEEP .0
4,22: GO TO 680
 660 NEXT l
 670 NEXT k
 680 NEXT i
 700 PRINT AT 21,1;"Mutare(a-p),
Rel(X),Cons(Z)    "
 710 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 715 PRINT AT 21,29;r$
 720 IF r$="x" THEN RESTORE : GO
 TO 10
 730 IF r$="z" THEN GO TO 1330
 740 FOR i=1 TO 16
 750 IF r$=l$(i) THEN GO TO 780
 760 NEXT i
 770 BEEP 1,-6: PRINT AT 21,19;"
 ": GO TO 710
 780 IF i<=4 THEN LET lin=i+1: G
O TO 1120
 790 IF i<=8 THEN LET lin=i-3: G
O TO 1020
 800 IF i<=12 THEN LET col=i-7:
GO TO 920
 810 LET col=i-11
 820 IF a$(6,col)<>"  " THEN GO
TO 770
 825 LET sum=0
 830 FOR i=1 TO 5
 840 IF a$(i,col)<>"  " THEN LET
 sum=sum+1
 850 NEXT i
 860 IF sum<>4 THEN GO TO 770
 870 FOR i=6 TO 2 STEP -1
 880 LET a$(i,col)=a$(i-1,col)
 885 LET a$(i-1,col)="  "
 890 PRINT AT 3*i-1,1+4*col; INV
ERSE 1;a$(i,col)
 895 IF a$(i,col)="  " THEN PRIN
T AT 3*i-1,1+4*col;"  "
 900 PRINT AT 3*(i-1)-1,1+4*col;
"  ": BEEP .02,12: BEEP .04,22
 910 NEXT i: GO TO 1220
 920 IF a$(1,col)<>"  " THEN GO
TO 770
 925 LET sum=0
 930 FOR i=2 TO 6
 940 IF a$(i,col)<>"  " THEN LET
 sum=sum+1
 950 NEXT i
 960 IF sum<>4 THEN GO TO 770
 970 FOR i=1 TO 5
 980 LET a$(i,col)=a$(i+1,col)
 985 LET a$(i+1,col)="  "
 990 PRINT AT 3*i-1,1+4*col; INV
ERSE 1;a$(i,col)
 995 IF a$(i,col)="  " THEN PRIN
T AT 3*i-1,1+4*col;"  "
1000 PRINT AT 3*(i+1)-1,1+4*col;
"  ": BEEP .02,12: BEEP .04,22
1010 NEXT i: GO TO 1220
1020 IF a$(lin,6)<>"  " THEN GO
TO 770
1025 LET sum=0
1030 FOR i=1 TO 5
1040 IF a$(lin,i)<>"  " THEN LET
 sum=sum+1
1050 NEXT i
1060 IF sum<>4 THEN GO TO 770
1070 FOR i=6 TO 2 STEP -1
1080 LET a$(lin,i)=a$(lin,i-1)
1085 LET a$(lin,i-1)="  "
1090 PRINT AT 3*lin-1,1+4*i; INV
ERSE 1;a$(lin,i)
1095 IF a$(lin,i)="  " THEN PRIN
T AT 3*lin-1,1+4*i;"  "
1100 PRINT AT 3*lin-1,1+4*(i-1);
"  ": BEEP .02,12: BEEP .04,22
1110 NEXT i: GO TO 1220
1120 IF a$(lin,1)<>"  " THEN GO
TO 770
1125 LET sum=0
1130 FOR i=2 TO 6
1140 IF a$(lin,i)<>"  " THEN LET
 sum=sum+1
1150 NEXT i
1160 IF sum<>4 THEN GO TO 770
1170 FOR i=1 TO 5
1180 LET a$(lin,i)=a$(lin,i+1)
1185 LET a$(lin,i+1)="  "
1190 PRINT AT 3*lin-1,1+4*i; INV
ERSE 1;a$(lin,i)
1195 IF a$(lin,i)="  " THEN PRIN
T AT 3*lin-1,1+4*i;"  "
1200 PRINT AT 3*lin-1,1+4*(i+1);
"  ": BEEP .02,12: BEEP .04,22
1210 NEXT i: GO TO 1220
1220 FOR i=1 TO 4: FOR j=1 TO 4
1230 IF a$(i+1,j+1)<>c$((i-1)*4+
j) THEN GO TO 700
1240 NEXT j: NEXT i
1250 PRINT AT 21,1;"FELICITARI -
 Ai reusit !!!     "
1260 FOR i=1 TO 3: FOR j=10 TO 3
0
1270 BEEP .02,RND*30+j
1280 NEXT j: NEXT i: PAUSE 40
1290 PRINT AT 21,1;"Alt joc (d/n
) ?                "
1300 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
1310 IF r$="d" THEN RESTORE : GO
 TO 10
1320 STOP
```

```
1330 PRINT AT 21,1;"Asteapta, te
 rog, putin          "
1340 DIM v$(16,2): LET loc=1
1350 FOR i=2 TO 5: FOR j=2 TO 5
1360 IF a$(i,j)="  " THEN GO TO
1480
1370 LET v$(loc)=a$(i,j): LET lo
c=loc+1
1380 NEXT j: NEXT i
1385 LET inv=0
1390 FOR i=1 TO 15
1400 FOR j=i+1 TO 16
1410 IF v$(i)>v$(j) THEN LET inv
=inv+1
1420 NEXT j: NEXT i
```

```
1425 IF inv=2*INT (inv/2) THEN G
O TO 1440
1430 PRINT AT 21,1;"Imposibil -
regret...            ": BEEP 1,-6: P
AUSE 60: GO TO 1290
1440 PRINT AT 21,1;"Posibil - Co
ntinui (d/n) ?    "
1450 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
1460 IF r$="d" THEN GO TO 700
1470 GO TO 1290
1480 PRINT AT 21,1;"Aranjeaza pi
esele in centru   "
1490 BEEP 1,-6: PAUSE 60: GO TO
700
```

Jocuri didactice

Vorbeam în introducerea lucrării despre revoluția pe care calculatoarele — cu precădere cele personale — sînt pe cale să o provoace în învățămînt. Folosirea calculatorului nu este un fenomen nou, primele manifestări datînd de prin anii '50. Pînă prin 1965, s-a mers mai ales pe linia realizării de programe ambițioase, încercînd să se substituie în general profesorului, în predare sau în evaluare. În unele locuri, demersul a fost exagerat, elevii fiind sufocați cu teste, repetitoare, chestionare puse pe calculator. Între timp, lucrurile au evoluat, atît din punct de vedere psiho-pedagogic, cît și informatic. Etapa calculatoarelor personale (cu posibilități grafice, inclusiv de animație și sonore) a adus în prim-plan programele de simulare și programele-joc. Primele sînt deosebit de utile în învățarea chimiei, fizicii, biologiei, matematicii, acolo unde se cere ilustrarea unor fenomene, rezultate, principii într-un mod cît mai cinematografic, dar cu posibilități de dialog (precizare de parametri, reluări, evaluări ale informației primite). Jocurile didactice pot fi folosite oriunde, cu condiția să fie jocuri, să includă adică o idee ludică vizibilă, bine adecvată subiectului. Realizarea unor asemenea jocuri nu este deloc simplă, pentru că trebuie menținută o balanță potrivită între caracterul de joc și cel didactic (« pericolele » fiind de a obține, la extreme, fie un joc propriu-zis de reflexe, fie un program repetitor sau de testare, neatractiv). Importanța acestor jocuri este însă incontestabilă, iar răspîndirea lor este din ce în ce mai mare, antrenînd și ajutînd ceea ce se cheamă un « învățămînt paralel » tot mai semnificativ, mai ales că pot beneficia de el nu numai elevii, ci oricine, indiferent de vîrstă. Se realizază, astfel, nu numai formarea și aprofundarea unor noțiuni, rezultate, abilități de calcul etc., ci și rememorarea acestora, în cazul celor care au absolvit de mai mult timp școala. După cum se vede, informatica nu mai aspiră la înlocuirea educatorului, ci la îmbogățirea panopliei mijloacelor de învățămînt, a materialelor didactice cu un instrument nou, deosebit de util, inteligent, atractiv.

Programele care urmează, alese din domenii școlare diferite, sînt doar o mostră de ceea ce se poate face în acest sens pe calculator, un mod de a îndemna cititorul în a-și valorifica fantezia prin realizarea unor seturi de asemenea programe, acoperitoare pentru capitole mai bogate din matematică, chimie, biologie sau alte obiecte de învățămînt.

VRĂJITORUL PORTOCALIU DIN ŢARA NUMERELOR

Este un joc cu ajutorul căruia copiii îşi pot exersa abilitatea de a efectua calcule aritmetice. Atragerea copiilor în acest joc didactic se realizează printr-un dialog hazliu şi antrenant cu un personaj de poveste, « Vrăjitorul portocaliu din ţara numerelor », care joacă rolul unui profesor. Chiar dacă uneori copiii încearcă să îl păcălească şi să-l înfurie pe Vrăjitor, de obicei fac acest lucru cu bună ştiinţă, rezultatul final fiind totdeauna acelaşi şi anume realizarea de exerciţii (mintale) cu numere.

La începutul jocului, Vrăjitorul se prezintă şi dă porunca: « ia un număr şi adună-l cu 3. Împarte rezultatul la 5, apoi înmulţeşte-l cu 8. Numărul obţinut îl împarţi la 5. Mai aduni 5, după aceea scazi 1. Care este rezultatul? » Jucătorul va introduce un număr, iar în funcţie de acesta Vrăjitorul va « ghici » care a fost numărul iniţial. Dacă totul se desfăşoară corect, adică calculele au fost bine efectuate şi jucătorul îşi recunoaşte numărul iniţial, Vrăjitorul portocaliu este mulţumit şi « zboară spre insuliţa sa ». Dacă apare o neconcordanţă între numărul ales de jucător şi cel indicat de Vrăjitor, acesta face o « demonstraţie » a calculului său. Dacă după această demonstraţie jucătorul s-a lămurit, Vrăjitorul portocaliu este mulţumit, dar dacă nu, înseamnă că jucătorul vrea să-l păcălească pe Vrăjitor şi acesta trimite asupra sa un « fulger cumplit », drept pedeapsă.

Observaţie: dacă rezultatul la care a ajuns jucătorul este un număr zecimal, atunci jucătorul va introduce acest număr cu **punct zecimal** (nu cu virgulă zecimală). De asemenea, calculele se vor efectua cu mare precizie (cu mai multe cifre zecimale, dacă este cazul) deoarece şi « puterea de calcul » a Vrăjitorului este mare (este, practic, puterea de calcul a limbajului BASIC).

Modificări posibile: pe aceeaşi idee, Vrăjitorul se poate transforma într-un profesor care să testeze capacitatea de a face calcule aritmetice pe diferite probe de dificultate şi, la sfîrşit, să pună o notă. De exemplu, să poată să testeze tabla înmulţirii.

Descrierea programului

10 — stabilirea atributelor şi apelul subrutinei (GO SUB 260) care realizează calculul duratei şi frecvenţei sunetelor (acestea sînt aleatoare în intervale alese în prealabil), emiterea unui sunet şi calculul culorii cu care se va afişa pe ecran (aleatoare). Modificarea valorii de la adresa

23692 (linia 290) asigură realizarea unui dialog fără apariţia pe ecran a mesajului «scroll?».

20–40 — prezentarea jocului şi a Vrăjitorului, însoţită şi de efecte sonore (apelarea subrutinei 260).

50–60 — începutul dialogului, marcat permanent de efecte sonore.

70–90 — prezentarea problemei.

100 — solicitarea rezultatului (variabila **b**).

110 — efectuarea calculului de către Vrăjitorul portocaliu (este un calcul invers) pentru aflarea numărului ales iniţial de către jucător. Acest calcul se memorează în variabila **c**.

120 — dialog.

130 — dacă jucătorul recunoaşte că numărul ales a fost «ghicit» de Vrăjitor, acesta «zboară» spre insuliţa lui (linia 240) şi apoi jocul ia sfîrşit.

140 — dacă jucătorul nu recunoaşte, Vrăjitorul îi solicită introducerea numărului ales iniţial (variabila **k**).

150 — calculul «mintal», de verificare, al Vrăjitorului. Cu numărul introdus de jucător va obţine rezultatul **j**.

160–180 — «demonstraţia» Vrăjitorului: dacă rezultatul obţinut de Vrăjitor (**j**) nu este egal cu cel introdus de jucător (**b**), înseamnă că acesta a greşit la calcule, iar dacă cele două rezultate sînt identice înseamnă că jucătorul a minţit cînd nu a recunoscut că Vrăjitorul a ghicit numărul.

180 — dacă acum jucătorul recunoaşte, Vrăjitorul pleacă (linia 240) şi jocul se termină.

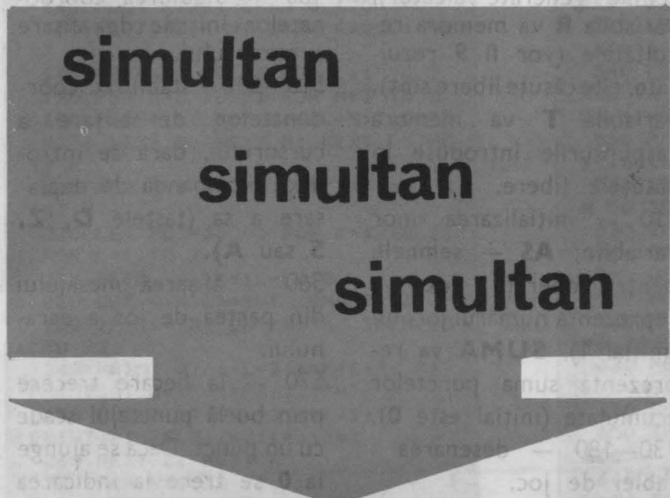
190 — jucătorul nu recunoaşte nici acum. Vrăjitorul se «înfurie».

200 — 230 — fulgerul.

```
  10 BORDER 5: PAPER 6: INK 1: C
LS : GO SUB 260
  20 PRINT AT 1,11;"VRAJITORUL":
 GO SUB 260
  30 PRINT ''''TAB 8;"Joc pentru
 copii": GO SUB 260: GO SUB 260:
 GO SUB 260
  40 PRINT '''"   Sint vrajitoru
l portocaliu          din tara nu
merelor .": GO SUB 260: GO SUB 2
60: PRINT "   Daca vrei sa devii
 ucenicul    meu,te voi pune la
incercare!": GO SUB 260: GO SUB
260
  50 INPUT AT 8,0;"    Esti gata
 ? (da/nu) ";a$: GO SUB 260: IF
a$="da" THEN GO TO 70
  60 GO SUB 260: PRINT '"  Daca
nu esti atent,te prefac    intr-
un norisor portocaliu !": GO SUB
 260: GO SUB 260
  70 PRINT '"  Ia un numar si ad
una-l cu 3 .": GO SUB 260
  80 PRINT "    Imparte rezultat
ul la 5 ,         apoi inmulteste
 cu 8 .    Numarul obtinut il i
mparti la 5."
  90 GO SUB 260: GO SUB 260: PRI
NT "Mai aduni 5, dupa aceea scaz
i 1.": GO SUB 260
 100 PRINT "   Spune-mi cit e re
zultatul ?": GO SUB 260: INPUT "
  rezultatul = ";b: GO SUB 260
 110 LET c=(b+1-5)*5/8*5-3: GO S
UB 260
 120 PRINT " Numarul tau a fost
 ";c: GO SUB 260: PRINT "    N-a
m dreptate ? (da/nu) ": GO SUB 2
60
 130 INPUT d$: GO SUB 260: IF d$
="da" THEN GO TO 240
 140 GO SUB 260: PRINT "   Care
era numarul initial ?": GO SUB 2
60: INPUT "   numarul = ";k: GO
SUB 260
 150 LET f=k+3: GO SUB 260: LET
g=f/5: LET h=g*8: GO SUB 260: LE
T i=h/5+5: LET j=i-1: GO SUB 260
 160 PRINT '"          Te crezi de
stept ?       Acum urmareste-ma
:": GO SUB 260: PRINT 'TAB 5;k;"
 plus 3 egal ";f;: GO SUB 260
 170 PRINT '"Acesta impartit la
5 egal ";g;'"Rezultatul inmultit
 cu 8 face ";h;'" Daca impartim
la 5 si adunam 5  obtinem ";i;"
care,minus 1 ,face ";j
 180 GO SUB 260: GO SUB 260: INP
UT "     Acum ma crezi ? (da/nu)"
;z$: GO SUB 260: IF z$="da" THEN
 GO TO 240
 190 PRINT " Ma faci nebun ?! HA
 ! Pe capul tau va cadea un fulg
er cumplit !        Ca pedeapsa
!!!": GO SUB 260: GO SUB 260: PA
USE 100
 200 FOR x=53 TO 45 STEP -1: GO
SUB 300: PRINT TAB x;"X(CAPS 88)
X": NEXT x
 210 PRINT TAB 44;"X(CAPS 88)XXX
XXX";: GO SUB 300: PRINT TAB 43;
"X(CAPS 888888)X";: GO SUB 300:
PRINT TAB 42;"XXXXXX(CAPS 88)X"
 220 FOR y=46 TO 43 STEP -1: GO
```

```
SUB 300: PRINT TAB y;"X<CAPS 88>
X": NEXT y: PRINT TAB 41;"X<CAPS
 88888>X";: GO SUB 300: PRINT TA
B 42;"X<CAPS 888>X";: GO SUB 300
: PRINT TAB 43;"X<CAPS 8>X";: GO
 SUB 300: PRINT TAB 44;"X"
 230 INK 0: GO SUB 260: PRINT '
FLASH 1;"    Alta data sa ma cre
zi !": GO SUB 260: GO SUB 260
 240 PRINT '"   Gata! Zbor spre i
nsulita mea!": FOR i=1 TO 6: BEE
```

```
P .1,i: NEXT i
 250 GO TO 320
 260 LET durata=(INT (RND*10+1))
/10
 270 LET frecventa=INT (RND*100-
50)+1
 280 BEEP durata,frecventa
 290 POKE 23692,255: RETURN
 300 LET ink=INT (RND*8): IF ink
=6 THEN GO TO 300
 310 INK ink: RETURN
```



Este un joc cu ajutorul căruia copiii își pot dezvolta deprinderea de a efectua rapid calcule matematice și, mai mult, de a identifica anumite numere care satisfac **simultan** mai multe egalități. Se poate spune că acest tip de joc va ajuta copiii (mai tîrziu) în problemele legate de rezolvarea sistemelor de ecuații și a **modelelor matematice.**

Pe ecran se afișează un caroiaj de 5×5 pătrate, din care 4 pătrate sînt blocate (colorate), iar 12 au înscris în ele semnele matematice corespunzătoare adunării, scăderii sau înmulțirii. Rămîn de completat de către jucător 9 pătrate cu cifrele care vor rezolva simultan ecuațiile reprezentate. Rezultatul fiecărei ecuații se găsește pe linia și pe coloana respectivă, în afara caroiajului. Un exemplu de situație inițială a jocului se poate urmări în fig. 16. În vederea rezolvării ecuațiilor se deplasează cursorul — ┌ — (el apare inițial pe prima poziție a caroiajului și clipește) cu ajutorul tastelor (**D** — cursor

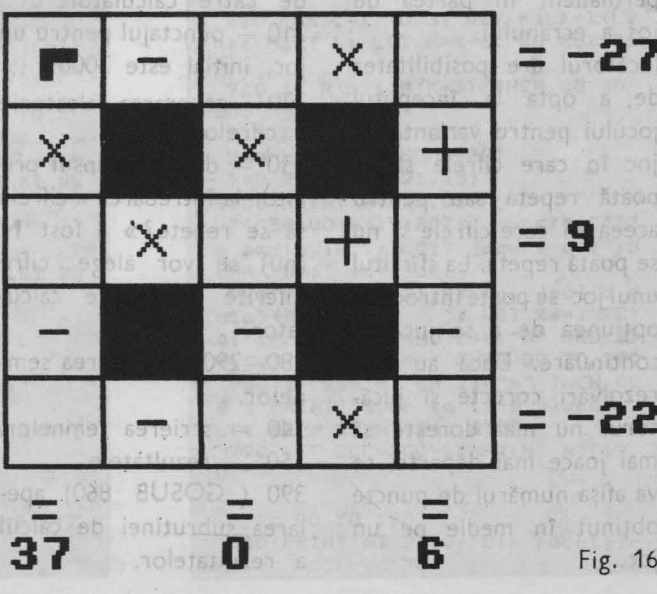


Fig. 16

dreapta, **S** — cursor stînga, **A** — cursor sus, **Z** — cursor jos) și se înscrie sau modifică cifre în pătratul pe care este poziționat cursorul, încercîndu-se prin modificarea făcută ca egalitățile să fie satisfăcute atît pe orizontală, cît și pe verticală. Scopul jocului este de a indica rezultatul exact (completarea pătratelor libere cu cifre corespunzătoare) într-un interval cît mai mic de timp. Inițial sînt 3000 de puncte. Numărul acestora va scădea proporțional cu timpul consumat pentru aflarea soluției. Dacă soluția nu este descoperită, se poate acționa tasta **K** și rezolvarea va apărea în colțul din dreapta-jos al fiecărui pătrat. Dacă soluția este bună, jucătorul va rămîne cu punctele din acel moment. Punctele se afișează permanent în partea de jos a ecranului.

Jucătorul are posibilitatea de a opta la începutul jocului pentru varianta de joc în care cifrele să se poată repeta sau pentru aceea în care cifrele să nu se poată repeta. La sfîrșitul unui joc se poate introduce opțiunea de a se juca în continuare. Dacă au fost rezolvări corecte și jucătorul nu mai dorește să mai joace mai departe, se va afișa numărul de puncte obținut în medie pe un joc.

Descrierea programului

80 — respingerea răspunsurilor nelegale.

100 — rezervarea de spațiu de memorie pentru variabile.

Variabila **C** va memora cifrele (vor fi cîte 3 cifre pe fiecare linie sau coloană); variabila **S** va memora semnele generate (vor fi 12 semne generate aleator), variabila **R** va memora rezultatele (vor fi 9 rezultate, cîte căsuțe libere sînt), variabila **T** va memora răspunsurile introduse în căsuțele libere.

110 — inițializarea unor variabile; **A$** — semnele pentru operații, **joc** va reprezenta numărul jocului (inițial 1), **SUMA** va reprezenta suma punctelor acumulate (inițial este **0**).

130—180 — desenarea tablei de joc.

205 — alegerea cifrelor de către calculator.

210 — punctajul pentru un joc, inițial este 3000.

220 — generarea aleatorie a cifrelor întregi.

230 — dacă răspunsul privitor la întrebarea «cifrele să se repete?» a fost **N** (nu) se vor alege cifre diferite de către calculator.

280—290 — alegerea semnelor.

340 — scrierea semnelor.

350 — rezultatele.

390 (GOSUB 860) apelarea subrutinei de calcul a rezultatelor.

870—900 calcul rezultate.

870–880 — calcul rezultate parțiale.

890–920 — calcul rezultate te **R**.

400 — scrierea rezultatelor

460 (GOSUB 360) apelarea subrutinei de calcul a rezultatelor și scrierea rezultatelor.

480—540 — bucla principală de introducere a datelor.

480 — stabilirea coordonatelor inițiale de afișare a cursorului.

510—530 — stabilirea coordonatelor de afișare a cursorului, dacă se introduce o comandă de deplasare a sa (tastele **D, Z, S** sau **A**).

560 — afișarea mesajului din partea de jos a ecranului.

570 — la fiecare trecere prin buclă punctajul scade cu un punct. Dacă se ajunge la **0** se trece la indicarea soluției (ca și cum s-ar fi dat un răspuns greșit).

580 — dacă se acționează tasta **K** se indică soluția.

610 — rezolvare terminată.

640 — se apelează subrutina de calcul a rezultatelor.

650 — dacă rezultatul nu coincide cu soluția, se va afișa mesajul și se va indica soluția.

730—750 — rezultatul pentru rezolvare corectă.

760 — joc nou?

810—840 — rezolvare corectă (soluția).

850 — joc nou?

```
  10 BORDER 7: PAPER 7: INK 0: C
LS
  70 PRINT AT 21,0;"Cifrele sa s
e repete(D/N)?": LET P$=INKEY$
  80 IF P$<>"D" AND P$<>"N" AND
P$<>"d" AND P$<>"n" THEN GO TO 7
0: CLS
  85 CLS
 100 RANDOMIZE : DIM C(3,3): DIM
 S(12): DIM I(3,3): DIM F(3): DI
M R(9): DIM T(9)
 110 LET A$="+-x": LET JOC=1: LE
T SUMA=0
 130 FOR K=48 TO 168 STEP 24
 135 BEEP .01,K/8
 140 PLOT K,39: DRAW 0,120
 150 PLOT 48,K-9: DRAW 120,0: NE
XT K
 160 FOR K=0 TO 1: FOR L=1 TO 2
 170 FOR M=5 TO 7: FOR N=3 TO 5
 180 PRINT AT 6*K+M,6*L+N;"<CAPS
 8>": NEXT N: NEXT M: NEXT L: NE
XT K
 205 FOR K=1 TO 9: LET T(K)=0: N
EXT K
 210 LET PCT=3000: FOR K=1 TO 3:
 FOR L=1 TO 3
 215 BEEP .01,K*10-L*15
 220 LET C(K,L)=INT (RND*9)+1
 230 IF P$="N" OR P$="n" THEN GO
 TO 930
 240 PRINT AT 6*L-3,6*K+1;"  ":
PRINT AT 3+(K-1)*6,22;"="
 250 PRINT AT 6*L-2,6*K+1;"  ":
PRINT AT 18,7+(K-1)*6;"="
 260 NEXT L: NEXT K
 280 FOR K=1 TO 12
 290 LET S(K)=INT (RND*3)+1: NEX
T K
 300 FOR L=1 TO 2: FOR K=1 TO 3
 305 BEEP .01,K*10-L*15
 310 PRINT AT 3+(K-1)*6,10+(L-1)
*6;A$(S(L+(K-1)*5)): NEXT K: NEX
T L
 320 FOR L=1 TO 3: FOR K=1 TO 2
 325 BEEP .01,L*15-K*10
 330 PRINT AT 6+(K-1)*6,7+(L-1)*
6;A$(S(2+(K-1)*5+L)): NEXT K: NE
XT L
 350 FOR K=1 TO 3
 355 BEEP .1,RND*30-40
 360 LET B1=S(1+(K-1)*5): LET B2
=S(2+(K-1)*5)
 370 FOR L=1 TO 3
 380 LET F(L)=C(K,L): NEXT L
 390 GO SUB 860
 395 PRINT AT 3+(K-1)*6,24;"

 400 PRINT AT 3+(K-1)*6,24;R(K):
 NEXT K
 410 FOR K=1 TO 3
 415 BEEP .1,RND*30-40
 420 LET B1=S(K+2): LET B2=S(K+7
)
 430 FOR L=1 TO 3
 440 LET F(L)=C(L,K): NEXT L: LE
T K=K+3
 450 GO SUB 860
 455 PRINT AT 20,11-(K-1)*6;"
"
 460 PRINT AT 20,11-(K-1)*6;R(K)
: LET K=K-3: NEXT K
 480 LET Y=4: LET X=8
 490 PRINT AT Y,X;"<CAPS 4>": PA
USE 5
 495 BEEP .01,10
 500 PRINT AT Y,X;" "
 510 LET X$=INKEY$
 520 LET X=X+6*((X$="d")*(X<16)-
(X$="s")*(X>9))
 530 LET Y=Y+6*((X$="z")*(Y<14)-
(X$="a")*(Y>9))
 540 LET VX=CODE X$-48: IF VX>0
AND VX<=9 THEN LET I((Y+2)/6,(X-
2)/6)=VX: PRINT AT Y-1,X-1;VX
 560 PRINT AT 21,0;"Puncte:";PCT
;".Gata-apasa K"
 570 LET PCT=PCT-1: IF PCT=0 THE
N GO TO 800
 580 IF X$="K" OR X$="k" THEN GO
 TO 600
 590 GO TO 490
 610 FOR K=1 TO 3
 620 LET B1=S(1+(K-1)*5): LET B2
=S(2+(K-1)*5)
 630 FOR L=1 TO 3: LET F(L)=I(K,
L): NEXT L: LET K=K+6
 640 GO SUB 860
 650 IF R(K)<>R(K-6) THEN GO TO
800
 660 LET K=K-6: NEXT K
 670 FOR K=1 TO 3
 680 LET B1=S(K+2): LET B2=S(K+7
)
 690 FOR L=1 TO 3: LET F(L)=I(L,
K): NEXT L: LET K=K+6: GO SUB 86
0
 710 IF R(K)<>R(K-3) THEN GO TO
800
 715 LET K=K-6: NEXT K
 730 LET SUMA=SUMA+PCT
 740 PRINT AT 21,25;"       "
 750 PRINT AT 17,0;"      ","
"'"BRAVO!Ai:";PCT;" puncte!"'"I
n medie,in ";JOC;" jocuri ai:";S
UMA/JOC;" pct/joc!"
 770 PRINT AT 21,0;"Mai joci o d
ata?(D/N)        ": LET Z$=INKEY
$: IF Z$<>"D" AND Z$<>"N" AND Z$
<>"d" AND Z$<>"n" THEN GO TO 770
 780 IF Z$="N" OR Z$="n" THEN CL
S : PRINT "Sper ca ti-a placut..
.": BEEP 2,RND*24: STOP
 790 LET JOC=JOC+1: PRINT AT 19,
0;"

  ": GO TO 190
 810 PRINT AT 21,0;"Din pacate,r
```

```
au.Iata rezolvarea:"
 820 FOR K=1 TO 3: FOR L=1 TO 3
 830 PRINT AT 4+(K-1)*6,8+(L-1)*
6;C(K,L)
 840 NEXT L: NEXT K
 845 FOR K=-30 TO 30 STEP 5: BEE
P .1,K: NEXT K
 850 GO TO 760
 870 IF B1=3 THEN LET W=F(1)*F(2
): GO TO 900
 880 IF B2=3 THEN LET W=F(2)*F(3
): GO TO 890
 885 GO TO 910
```

```
 890 LET R(K)=(F(1)+W*((B1=1)-(B
1=2))): RETURN
 900 LET R(K)=(W+F(3)*((B2=1)-(B
2=2)))*(B2<>3)+(F(3)*W)*(B2=3):
RETURN
 910 LET W=F(1)+F(2)*((B1=1)-(B1
=2))
 920 LET R(K)=W+F(3)*((B2=1)-(B2
=2)): RETURN
 940 IF T(C(K,L))=1 THEN GO TO 2
20
 950 LET T(C(K,L))=1: PRINT AT 1
3,0;"SCRIE"'"CIFRE"'"DIFE-"'"RIT
E!": GO TO 240
```

A N I M A L E

Un joc pe care copiii îl joacă deseori între ei, sub numele de « Ghicește animalul ». Un jucător se gîndește la un animal, iar ceilalți încearcă să-l « ghicească » punînd întrebări și obținînd astfel informațiile necesare pentru identificarea lui. Cu cît întrebările vor fi puse mai inteligent, restrîngînd mai rapid clasa din care face parte animalul, cu atît numărul de încercări din care acesta va fi identificat va fi mai mic. Caracterul instructiv al jocului rezultă din învățarea, pe această cale, a unor specii de animale precum și a caracteristicilor importante prin care animalele se deosebesc între ele (mediul și locul în care trăiesc, cum se înmulțesc, dacă este domestic, caracteristici fizice definitorii etc).

În jocul simulat pe calculator, acesta are rolul de a « ghici » animalul. Jucătorul va fi deci invitat să se gîndească la un animal. Calculatorul pune apoi diverse întrebări (trăiește la noi?, trăiește în apă?, are blană?, are pene?, este domestic? etc), la care jucătorul va răspunde prin « da » sau « nu » (se poate răspunde și numai prin indicarea primei litere a răspunsului: « d » sau « n »). După mai multe întrebări, calculatorul va indica un animal. Dacă răspunsul a fost corect (este chiar animalul la care se gîndește jucătorul), în urma opțiunii jucătorului, se va putea începe (sau nu) alt joc. Dacă răspunsul nu a fost corect, atunci calculatorul va cere jucătorului să formuleze o întrebare care deosebește animalul indicat de animalul

care trebuia ghicit. Astfel programul are o caracteristică foarte importantă: **el poate învăța lucruri noi,** adică își adaugă animalul care trebuia ghicit printre animalele pe care le cunoștea deja. În acest fel, programul își poate îmbogăți neîncetat zestrea de cunoștințe ajungînd în timp să reprezinte pentru un jucător un adevărat atlas zoologic. Dacă la întrebarea de continuare a jocului, jucătorul răspunde cu « nu » (sau « n »), calculatorul va întreba dacă se dorește înregistrarea programului (jocului) cu toate animalele pe care le cunoaște. Dacă jucătorul răspunde afirmativ, va apare mesajul « Start cass and press any key »: se va poziționa caseta magnetică în dreptul unei porțiuni neînregistrate, se vor acționa clapele casetofonului pentru înregistrare și apoi orice tastă a calculatorului. În acest fel programul se va înregistra (salva) din nou, dar cînd se va încărca cu altă ocazie pentru a se juca jocul, va « cunoaște » toate animalele pe care le știa în momentul înregistrării.

Două aspecte noi intervin în acest joc față de jocul tradițional (fără calculator). Primul este legat de achiziționarea de cunoștințe și, pe această bază, a dezvoltării deprinderii de arhivare și sistematizare a informațiilor (la copii), iar al doilea este legat de faptul că jucătorul interpretează în timpul jocului atît un rol (aparent) pasiv (specific jucătorului care se gîndește la un animal), cît și un rol activ (specific jucătorului care « ghicește » animalul). Într-adevăr, în momentul în care jucătorul este solicitat de către calculator să pună o întrebare care face o deosebire esențială între două animale, rolul său devine activ, semănînd într-o mare măsură cu rolul jucat pînă atunci de calculator. Nu recomandăm modificări ale jocului (cu toate că s-ar putea realiza astfel de jocuri pe diferite specii de animale: pești, păsări, animale sălbatice, animale domestice, mamifere etc), în schimb recomandăm adăugarea de cît mai multe animale în baza de cunoștințe a programului (programul va putea « înmagazina » pînă la 500 de animale).

Descrierea programului

În realizarea programului s-au folosit tehnici specifice inteligenței artificiale. Astfel, pentru structurarea datelor (care constau din nume de animale sau întrebări) s-a utilizat un **arbore de tip binar** (vezi fig. 17). Acesta este format din noduri, fiecărui nod fiindu-i asociată o întrebare (dacă este un nod intermediar) și posibilitatea unei ramificații corespunzătoare răspunsului pozitiv sau negativ

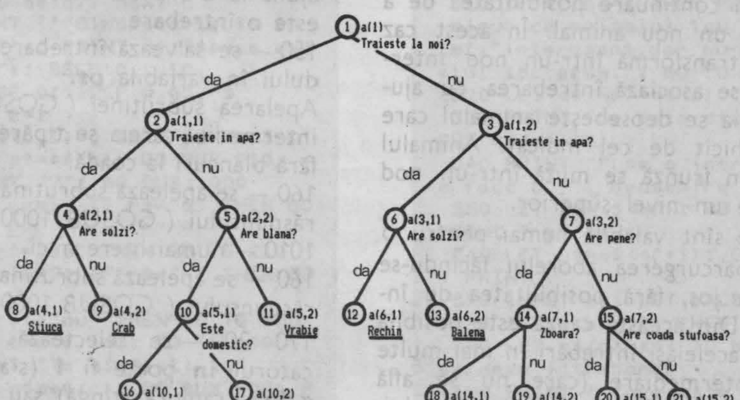


Fig. 17

la întrebare (de unde şi denumirea de arbore binar). Arborele porneşte de la un nod iniţial (rădăcina arborelui) şi se continuă cu noduri moştenitoare. Nodurile sînt numerotate: rădăcina este nodul 1, pe următorul nivel (primii moştenitori) sînt nodurile 2 şi 3, iar pe următorul nivel nodurile 4, 5, 6 şi 7 (moştenitorii nodurilor 2 şi respectiv 3) şi aşa mai departe. În program, arborele este descris în variabila **a(nq, j)** (vezi linia 15), în care **nq** reprezintă numărul nodului «tată», iar **j** poate fi **1** sau **2**, după cum nodul este pe ramura din partea stîngă sau pe ramura din partea dreaptă. Modul de creştere a arborelui este următorul: dacă nodul **i** este un nod «tată», atunci moştenitorii săi vor fi modurile **k** şi **l**, iar acestea vor avea asociate informaţiile (pot fi întrebări sau animale) **a (i, 1)** pentru nodul moştenitor de pe ramura stîngă şi **a (i, 2)** pentru nodul moştenitor de pe ramura dreaptă.

Dacă un nod are moştenitori, atunci el va reprezenta în arbore un nod frunză. Aceste noduri aŭ ca informaţii asociate animale iar informaţiile asociate moştenitorilor au valoarea **0**.

Programul parcurge arborele de sus în jos (de la rădăcină spre frunze) şi, prin intermediul răspunsurilor la întrebări, va ajunge la o frunză (deci la un animal), existînd în continuare posibilitatea de a se adăuga un nou animal. În acest caz frunza se transformă într-un nod intermediar. I se asociază întrebarea cu ajutorul căruia se deosebeşte animalul care trebuia ghicit de cel indicat. Animalul care era în frunză se mută într-un nod frunză, pe un nivel superior.

Întrebările sînt valabile numai pentru o ramură, parcurgerea aborelui făcîndu-se de sus în jos, fără posibilitatea de întoarcere. Din această cauză este posibilă existenţa aceleiaşi întrebări în mai multe noduri intermediare (care nu se află pe aceeaşi ramură) în cadrul arborelui, cum se poate observa şi în construcţia programului prezentat, de exemplu, cu întrebarea: «trăieşte în apă?». Acest lucru este logic, deoarece atît animalele care trăiesc la noi, cît şi cele care trăiesc în alte locuri pot avea (sau nu) ca mediu de viaţă, apa.

10 — variabila **nq** reprezintă numărul maxim de întrebări şi animale pe care le poate «ţine minte» programul. În exemplul nostru s-a luat un număr «rotund» (500), dar acesta se poate modifica pînă la 635.

15 — rezervarea de spaţiu de memorie pentru întrebări şi nume de animale. Variabile: întrebarea pentru modul **i** se află în **q$(i)**. Lungimea maximă a întrebării poate fi de 50 de caractere; **a(i, 1)** pointer stînga pentru nodul **i**; **a(i, 2)** pointer dreapta pentru nodul **i**; **r$(1)** răspunsul de la tastatură, **da** sau **nu**.

20 — Variabila **gf** reprezintă numărul iniţial de noduri ocupate în arbore. Aşa cum este construit programul, se porneşte cu 21 de întrebări şi răspunsuri (10 întrebări şi 11 animale, totdeauna numărul de animale — frunze — fiind cu o unitate mai mare decît numărul de noduri intermediare).

30—50 — se încarcă nodurile (se citesc întrebările şi pointerii — din stînga şi din dreapta).

60—70 — se încarcă frunzele

110 — începerea jocului.

130 — variabila **c** reprezintă numărul nodului. Se începe parcurgerea arborelui de la rădăcină (cu o întrebare).

140 — dacă **a(c, 1) = 0** înseamnă că s-a ajuns la o frunză, dacă nu, înseamnă că este o întrebare.

150 — se salvează întrebarea asociată nodului în variabila **ps**.

Apelarea subrutinei (GOSUB 910) prin intermediul căreia se tipăreşte întrebarea fără blancuri la coadă.

160 — se apelează subrutina pentru citirea răspunsului (GOSUB 1000).

1010 — numai litere mici.

160 — se apelează subrutina pentru citirea răspunsului (GOSUB 1000).

170—200 — se selectează ramura. Indicatorul **in** poate fi **1** (s-a răspuns prin «**da**», ramura stîngă) sau **2** (s-a răspuns prin «**nu**», ramura dreaptă).

210 — se trece la nodul următor şi procedeul se repetă (GO TO 140).

310 — s-a ajuns într-o frunză, deci la un animal.

320 — se tipărește animalul la care s-a ajuns (GO SUB 900 — fără spații la coadă).

330 (GO SUB 1000) — se apelează subrutina pentru citirea răspunsului.

340—350 — dacă s-a ghicit, se trece la opțiunea pentru alt joc sau salvarea programului (800).

360—370 — dacă nu s-a ghicit, se trece la inserarea unui nou animal prin lungirea arborelui.

510 — dacă nu mai este loc pentru un nou animal, se trece la opțiunile pentru sfîrșit joc (800).

520 — mutarea vechiului animal.

530 — introducerea numelui noului animal; numărul de noduri ocupate din arbore (**qf**) crește cu o unitate.

590 — inserează întrebarea.

630—680 — în funcție de răspuns, ramurile se schimbă între ele.

710 — inițializează efectiv nodurile, reface răspunsurile.

720 — rezervă următorul spațiu liber pentru animal.

725 — alegerea aleatorie a unuia din cele cinci mesaje.

730—770 — mesaje de afișat.

810 — opțiuni de final de joc.

816 — șterge mesajul de pe liniile din partea de jos a ecranului.

2010—2110 — datele privind întrebările și numele animalelor.

2010— nodul **1** și moștenitorii săi (nodul **2** și nodul **3**).

2020 — nodul **2** și moștenitorii săi (nodul **4** și nodul **5**).

2080 — nodul **15** și moștenitorii săi (**20** și **21**).

2120 — numele animalelor (din noduri, frunze).

```
 10 LET nq=100
 15 DIM q$(nq,50): DIM a(nq,2):
 DIM r$(1)
 20 LET qf=22
 30 FOR n=1 TO qf/2-1
 40 READ q$(n): READ a(n,1): RE
AD a(n,2)
 50 NEXT n
 60 FOR n=n TO qf-1
 70 READ q$(n): NEXT n
 110 PRINT ''"Gindeste-te la un
animal."'';: PRINT #1;"Apasa ori
ce tasta";: BEEP 0.1,10
 120 PAUSE 0: BEEP 0.05,-5
 130 LET c=1
 140 IF a(c,1)=0 THEN GO TO 300
 150 LET p$=q$(c): GO SUB 910
 160 PRINT "?": GO SUB 1000
 170 LET in=1: IF r$="d" THEN GO
 TO 210
 180 IF r$="da" THEN GO TO 210
 190 LET in=2: IF r$="n" THEN GO
 TO 210
 200 IF r$<>"nu" THEN GO TO 150
 210 LET c=a(c,in): GO TO 140
 310 PRINT "Te gindesti la"
 320 LET p$=q$(c): GO SUB 900: P
RINT "?"
 330 GO SUB 1000
 340 IF r$="d" OR r$="da" THEN G
O TO 400
 350 IF r$="D" OR r$="DA" THEN G
O TO 400
```

```
 360 IF r$="n" OR r$="nu" THEN G
O TO 500
 370 IF r$="N" OR r$="NU" THEN G
O TO 500
 380 PRINT "Raspunde-mi clar cin
d vorbesc "'"cu tine!?"': GO TO
300
 410 PRINT "M-am gindit atita!":
 GO TO 800
 510 IF qf>nq-1 THEN PRINT "Sint
 sigur ca animalul tau este foar
te","interesant,dar nu mai am de
stul loc acum.": GO TO 800
 520 LET q$(qf)=q$(c)
 530 PRINT "Ce este atunci?": IN
PUT q$(qf+1)
 540 PRINT "Pune o intrebare car
e face o"'" deosebire intre ";
 550 LET p$=q$(qf): GO SUB 900:
PRINT " si ";
 560 LET p$=q$(qf+1): GO SUB 900
: PRINT
 570 INPUT s$: LET b=LEN s$
 580 IF s$(b)="?" THEN LET b=b-1
 590 LET q$(c)=s$( TO b): REM in
sereaza intrebarea
 600 PRINT '"Care este raspunsul
 la"
 610 LET p$=q$(qf+1): GO SUB 900
: PRINT "?"
 620 GO SUB 1000
 630 LET in=1: LET io=2
 640 IF r$="d" OR r$="da" THEN G
O TO 700
```

```
 650 IF r$="D" OR r$="DA" THEN G
O TO 700
 660 LET in=2: LET io=1
 670 IF r$="n" OR r$="nu" THEN G
O TO 700
 680 IF r$="N" OR r$="Nu" THEN G
O TO 700
 690 PRINT "Incearca sa fi mai p
recis!"': GO TO 600
 710 LET a(c,in)=qf+1: LET a(c,i
o)=qf
 720 LET qf=qf+2
 725 GO TO 730+10*INT (RND*5)
 730 PRINT "Asta ma ameteste. ":
 GO TO 800
 740 PRINT "Nu m-as fi gindit !"
: GO TO 800
 750 PRINT "Foarte interesant.":
 GO TO 800
 760 PRINT "Daca spui tu sigur e
asa.": GO TO 800
 770 PRINT "Aflu lucruri noi .":
 GO TO 800
 810 LET r$=INKEY$: PRINT #1;"In
ca un joc ?";: BEEP 0.5,2: LET r
$=INKEY$: BEEP 0.2,6: PAUSE 50:
LET r$=INKEY$: PAUSE 50
 815 LET r$=INKEY$: PRINT : PRIN
T r$: PRINT : IF r$="" THEN GO T
O 815
 816 INPUT ""
 820 IF r$<>"n" AND r$<>"N" THEN
 GO TO 100
 840 INPUT "Vrei sa salvezi prog
ramul cu    toate animalele? ";r
$
 850 IF r$(1)="d" THEN INPUT "Nu
mele programului: ";r$: SAVE r$
LINE 100
 860 STOP
 905 PRINT " ";
 910 FOR n=50 TO 1 STEP -1
 920 IF p$(n)<>" " THEN GO TO 93
5
 930 NEXT n
 935 IF p$(1)>"`" THEN LET p$(1)
=CHR$ (CODE p$(1)-32)
 940 PRINT p$( TO n);: RETURN
1010 POKE 23658,16: BEEP 0.05,20
: INPUT LINE r$: IF r$="" THEN G
O TO 1010
1020 LET r$=r$(1): RETURN
2010 DATA "Traieste la noi",2,3
2020 DATA "Traieste in apa",4,5
2030 DATA "Traieste in apa",6,7
2040 DATA "Are solzi",11,12
2050 DATA "Are blana",8,13
2060 DATA "Are solzi",14,15
2070 DATA "Are pene",9,10
2080 DATA "Este un animal domest
ic",20,21
2090 DATA "Zboara",16,17
2100 DATA "Are coada stufoasa",1
8,19
2110 DATA "o stiuca","un crab","
o vrabie","un rechin","o balena"
,"o agvila","un pinguin","un sco
ncs","un elefant","o vaca","o vu
lpe"
```

DIPO — MODELUL EVOLUȚIEI UNEI POPULAȚII

Este un joc renumit, prin care se simulează evoluția unei colonii (populații) de celule după anumite legi (reguli). Evoluția unei celule va fi marcată de următoarele evenimente posibile:

a) celula vie supraviețuiește;

b) celula vie moare;

c) s-a generat o nouă celulă vie.

Cele trei stări se determină după următoarele reguli:

1. O celulă supraviețuiește dacă are 2 sau 3 celule vii vecine.

2. O celulă moare dacă nu are alte celule vii vecine (prin izolare) sau dacă are 4 sau mai multe celule vii vecine (prin suprapopulare).

3. Naștere — în fiecare locație adiacentă la 3 celule vii vecine se va naște o celulă vie în generația următoare. Acest model de creștere a fost prima oară studiat de Conway. De aceea, uneori această simulare se mai numește **« Universul lui Conway »**.

Modelul de creștere implică inexistența unor structuri inițiale care să conducă la o creștere fără limite a populației și existența unor structuri simple care se dezvoltă trecînd prin mai multe generații înainte de a ajunge la un sfîrșit în 3 moduri posibile: dispariția populației, stabilizarea structurii sau intrarea într-o fază oscilatorie în care se repetă două sau mai multe structuri de generații.

Simularea evoluției cu ajutorul calculatorului se efectuează pe o zonă (grilă) de N×M locații pe care se generează celule vii (colonia) conform datelor introduse de utilizator. O locație poate avea două stări: conține sau nu o celulă vie. Din starea inițială, în baza legilor geneticii descrise, se ajunge la generația următoare. Jocul se va termina atunci cînd populația dispare în totalitate și nu mai este nici un supraviețuitor, fapt semnalat de calculator prin afișarea unui mesaj. Pentru celelalte două cazuri posibile (stabilizare structură sau ciclare) rămîne ca utilizatorul să le identifice și să întrerupă simularea cînd va crede de cuviință (cu BREAK).

În jocul prezentat se cere inițial introducerea limitelor pentru fixarea cadrului grilei de N×M locații: mai întîi limita la stînga și la dreapta (de la 2 la 19), apoi limita de sus și de jos (de la 2 la 19). După introducerea primei limite se va acționa CR apoi se va introduce a 2-a limită. Apoi se începe inițializarea primei generații de către jucător. Acest lucru se realizează conform indicațiilor afișate pe ecran și anume:

— pentru deplasarea cursorului (notat cu semnul "X") în vederea poziționării acestuia în locația în care se dorește marcarea unei celule, se folosesc tastele: **5** (deplasare stînga); **6** (deplasare jos); **7** (deplasare sus) și **8** (deplasare dreapta).

— pentru marcarea unei celule într-o locație a grilei (după ce cursorul a fost poziționat în acea locație): tasta **2.**

— pentru ștergerea unei celule dintr-o locație a grilei (după ce cursorul a fost poziționat în acea locație, deasupra celulei existente): tasta **0.**

— pentru indicarea sfîrși-

tului inițializării generației tasta ENTER (CR).

După terminarea inițializării, pentru începerea simulării se va acționa orice tastă. Va apărea grila cu cursorul (semnul « X ») pe prima locație a grilei. Cu ajutorul comenzilor indicate se va inițializa de către jucător prima generație, după care va începe simularea: celulele generației actuale vor fi colorate în verde, celulele care se nasc vor fi colorate în roșu iar celulele care mor, în negru. Apoi se vor genera următoarele generații, una după alta. Trebuie avut însă răbdare deoarece intervalul de timp dintre două generații poate fi destul de mare din cauza numeroaselor calcule care se efectuează, precum și a colorării celulelor.

În funcție de modelul (desenul) de la care se pornește inițial, structurile se pot stabiliza (în cazul în care populația nu dispare) în niște modele interesante (om, astronaut, iepure, pălărie etc.). Se poate efectua un studiu prin care să se identifice modelele inițiale care conduc la modele interesante.

Modificări propuse. Deoarece intervalul de timp între generații este destul de mare, se poate renunța la afișarea (în culori) a celulelor care se nasc și a celor care mor, mai ales că pentru un televizor/monitor alb-negru această reprezentare nu evidențiază clar celulele generației actuale de cele care se vor naște sau vor muri. În acest caz se poate renunța definitiv la culori: va apărea reprezentată, de fiecare dată, numai generația actuală (nu se vor mai evidenția celulele care se vor naște sau care mor). Această modificare va simplifica programul.

Se mai propune o modificare: în realitate, populațiile de celule nu sînt limitate de un cadru. Ce se întîmplă atunci, însă, cu celulele care ar putea să se nască în locații care nu sînt pe grilă? Mărirea grilei la fiecare generație nu este o soluție pentru ecranul calculatorului care este și el limitat. În acest caz, o soluție posibilă ar fi ca celulele care se pot naște și « ies » din grilă să fie reintroduse în grilă pe partea opusă față de aceea prin care au ieșit. Și acum pot apărea probleme, în special, cu celulele care « ies » prin colțurile grilei.

Descrierea programului

15 (GO SUB 700) — apelarea subrutinei de creare a caracterului grafic utilizator pentru celule (corespunzător tastei **A**).

700 — datele pentru motivul (caracterul) celulă.

710—750 — crearea motivului.

30 — fixarea limitelor grilei pe orizontală.

35 — precauții pentru limite care nu sînt întregi.

40 — schimbarea limitei stînga cu cea pentru dreapta, dacă acestea au fost introduse în ordine inversă.

45—50 — refuzarea valorilor pentru limite care nu se încadrează în intervalul cerut.

60—80 — introducerea și verificările privind limitele pe verticală.

82 — ștergerea mesajului afișat la linia 25.

90 — variabila **ng** va reprezenta numărul generației (inițial va fi prima generație).

110 — rezervarea de spațiu de memorie pentru fișierele care reprezintă indivizii (celulele) unei generații: **a** reprezintă vechea generație iar **n** reprezintă noua generație.

120—150 — ciclu de inițializare pentru prima generație: 7 semnifică absența unei celule (culoare albă).

170 — afișarea modului de lucru pentru constituirea primei generații.

177 — pauză foarte lungă care poate fi întreruptă dacă se urmărește indicația din linia 175.

180 — celulele care se nasc se vor desena cu roșu.

182 (GOSUB 800) — apelarea subrutinei de creare a unui caracter grafic (corespunzător tastei **B**) și de desenare cu ajutorul lui a grilei.

800 — datele pentru crearea motivului.

860—890 — ciclu pentru desenarea grilei.

190—200 — poziție inițială a semnului «X», indicînd căsuța gata a fi umplută.

205 — mărimea inițială a deplasamentului vertical și orizontal al semnului « X ».

210 — memorarea ultimei taste acționate.

220 — cazul umplerii unei căsuțe cu pregătirea deplasamentului orizontal al semnului « X». Caracterul grafic se obține acționînd tasta **A** în modul grafic.

225 — ștergerea unei căsuțe umplute din greșeală; aceeași pregătire ca și la linia 220.

230 — terminarea umplerii grilei.

235 — dacă s-a acționat comandă (tastă) invalidă.

245, 246, 247, 248 — deplasare viitoare (dreapta, jos, sus, stînga).

250, 260 — calculul viitoarei poziții a semnului « X ».

270 — dacă viitoarea poziție va fi în afara grilei (stînga sau sus) se refuză mișcarea.

280 — dacă viitoarea poziție va fi la sfîrșitul liniei, semnul va trebui trecut la începutul liniei următoare.

290 — dacă viitoarea poziție va fi în afara grilei (sus sau jos) se refuză mișcarea.

295 — ștergerea semnului « X », afișarea caracterului grafic corespunzător unei pătrățele (tasta **B** în modul grafic).

297 — afișarea unei căsuțe pline în cazul în care ea a fost parțial ștearsă cînd semnul « X » a fost șters.

300, 310 — reîntoarcere la linia 200 pentru afișarea semnului « X » în noul său loc.

355 — valoarea inițială a numărului total de indivizi (celule) ai unei generații **nt**.

362 (GO SUB 1000) — apelarea subrutinei de trasare a cadrului de viață a unei generații.

1010—1030 — trasare cadru sus și jos.

1020 — se folosește caracterul grafic corespunzător tastei **3** împreună cu **CAPS SHIFT** în modul grafic.

1025 — se folosește caracterul grafic corespunzător tastei **3** în modul grafic.

1040—1070 — trasare cadru dreapta și stînga.

1050 — se folosește caracterul grafic corespunzător tastei **5** în modul grafic.

1060 — se folosește caracterul grafic corespunzător tastei **5** împreună cu **CAPS SHIFT** în modul grafic.

365—385 — ciclu pentru afișarea unei generații și recensămîntul populației.

395 — numărul total al indivizilor populației este **0;** populația se stinge.

400—550 — ciclu pentru studiul unei generații: determinarea nașterilor, a deceselor și actualizarea afișajului (verde pentru nașteri și negru pentru decese).

430—460 — ciclu pentru determinarea numărului de celule vecine (**nc**).

470 — corecție pentru evitarea numărării celulei studiate ca propriul său vecin.

485 — decizie privind viitorul unei celule: supraviețuire sau deces. Sînt 9 cazuri posibile în funcție de numărul de vecini (de la 0 la 8).

490, 492 — determinarea culorilor: verde pentru generația actuală, roșu pentru generația viitoare.

Dacă nu se vor naște celule, culoarea va fi albă.

500—508 — studiul celor 9 cazuri.

530 — actualizarea afișajului.

580—610 — noua generație devine « vechea » generație.

615—620 — se trece la studiul generației noi (actuale).

640—660 — sfîrșit.

```
  10 BORDER 6: PAPER 7: INK 0
  15 GO SUB 700
  20 PRINT "Jocul viata"
  25 PRINT AT 2,2;"Limitele cadr
ului?"
  30 BEEP 0.2,20: INPUT "Limita
stinga si dreapta (2 la  19) ";c
min,cmax
  35 LET cmin=INT cmin: LET cmax
=INT cmax
  40 IF cmin>cmax THEN LET caux=
cmin: LET cmin=cmax: LET cmax=ca
ux
```

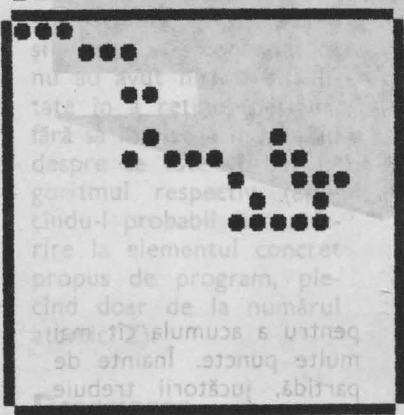
```
  45 IF cmin<2 OR cmax>19 THEN G
O TO 30
  50 IF cmin=cmax THEN GO TO 30
  60 BEEP 0.2,20: INPUT "Limita
sus si jos (2 la 19) ";lmin,lmax
  65 LET lmin=INT lmin: LET lmax
=INT lmax
  70 IF lmin>lmax THEN LET laux=
lmin: LET lmin=lmax: LET lmax=la
ux
  75 IF lmin<2 OR lmax>19 THEN G
O TO 60
  80 IF lmin=lmax THEN GO TO 600
```

```
 82 PRINT AT 2,2;"
 "
 90 LET ng=1
 100 PRINT AT 2,2;"Initializarea
 primei generatii"
 110 DIM a(lmax+1,cmax+1): DIM n
(lmax+1,cmax+1)
 120 FOR l=lmin-1 TO lmax+1
 130 FOR c=cmin-1 TO cmax+1
 140 LET a(l,c)=7: LET n(l,c)=7
 150 NEXT c: NEXT l
 170 PRINT AT 5,2;"Introduceti g
eneratia :        deplasare  tast
ele 5 6 7 8       celula     tast
a 2               stergere   tast
a 0               sfirsit    tast
a ENTER"
 175 PRINT AT 10,1;"Apasati o ta
sta pentru incepere"
 177 PAUSE 10000
 180 CLS : INK 2
 182 GO SUB 800
 185 BEEP 0.2,20
 190 LET l=lmin: LET c=cmin
 200 PRINT AT l,c;"X"
 205 LET dl=0: LET dc=0
 210 LET t$=INKEY$
 220 IF t$="2" THEN LET a(l,c)=2
: BEEP 0.2,10: PRINT OVER 1;AT l
,c;"<a>": LET dc=1: GO TO 250
 225 IF t$="0" THEN LET a(l,c)=7
: PRINT AT l,c;" ": LET dc=1: GO
 TO 250
 230 IF t$=CHR$ 13 THEN GO TO 35
0
 235 IF t$<"5" OR t$>"8" THEN GO
 TO 200
 240 GO TO 240+CODE t$-48
 245 LET dc=-1: GO TO 250
 246 LET dl=1: GO TO 250
 247 LET dl=-1: GO TO 250
 248 LET dc=1: GO TO 250
 250 LET lf=l+dl
 260 LET cf=c+dc
 270 IF cf<cmin THEN GO TO 200
 280 IF cf>cmax THEN LET cf=cmin
: LET lf=lf+1
 290 IF lf<lmin OR lf>lmax THEN
GO TO 200
 295 PRINT INVERSE 1;AT l,c;"X":
 PRINT INVERSE 0;AT l,c;"<b>"
 297 IF a(l,c)=2 THEN PRINT AT l
,c;"<a>"
 300 LET l=lf: LET c=cf
 310 GO TO 200
 350 CLS : INK 2
 355 LET nt=0
 360 PRINT AT 0,2;"generatia num
arul ";ng
 362 GO SUB 1000
 365 FOR l=lmin TO lmax
 370 FOR c=cmin TO cmax
 380 IF a(l,c)=2 THEN PRINT AT l
,c;"<a>": LET nt=nt+1
 385 NEXT c: NEXT l
 395 IF nt=0 THEN GO TO 640
 400 FOR l=lmin TO lmax
 410 FOR c=cmin TO cmax
 420 LET nc=0
 430 FOR i=l-1 TO l+1
 440 FOR j=c-1 TO c+1
 450 IF a(i,j)=2 THEN LET nc=nc+
1
 460 NEXT j: NEXT i
 470 IF a(l,c)=2 THEN LET nc=nc-
1
 485 IF a(l,c)=2 THEN GO TO 500+
nc
 490 LET cul=7
 492 IF nc=3 THEN LET cul=4: LET
 n(l,c)=2
 495 GO TO 530
 500 LET cul=0: LET n(l,c)=7: GO
 TO 530
 501 LET cul=0: LET n(l,c)=7: GO
 TO 530
 502 LET cul=2: LET n(l,c)=2: GO
 TO 530
 503 LET cul=2: LET n(l,c)=2: GO
 TO 530
 504 LET cul=0: LET n(l,c)=7: GO
 TO 530
 505 LET cul=0: LET n(l,c)=7: GO
 TO 530
 506 LET cul=0: LET n(l,c)=7: GO
 TO 530
 507 LET cul=0: LET n(l,c)=7: GO
 TO 530
 508 LET cul=0: LET n(l,c)=7
 530 PRINT INK cul;AT l,c;"<a>"
 550 NEXT c: NEXT l
 580 FOR l=lmin TO lmax
 590 FOR c=cmin TO cmax
 600 LET a(l,c)=n(l,c)
 610 NEXT c: NEXT l
 615 LET ng=ng+1
 620 GO TO 350
 640 PRINT AT 20,0;"Nu mai e nic
i un supravietuitor!"
 650 BEEP 0.2,13: BEEP 0.2,16
 660 STOP
 700 DATA 0,60,126,126,126,126,6
0,0
 710 FOR i=0 TO 7
 720 READ a
 730 POKE USR CHR$ 65+i,a
 740 NEXT i
 750 RETURN
 800 DATA 255,129,129,129,129,12
9,129,255
 810 FOR i=0 TO 7
 820 READ a
 830 POKE USR CHR$ 66+i,a
 840 NEXT i
 860 FOR l=lmin TO lmax
 870 FOR c=cmin TO cmax
 880 PRINT AT l,c;"<b>"
 890 NEXT c: NEXT l
```

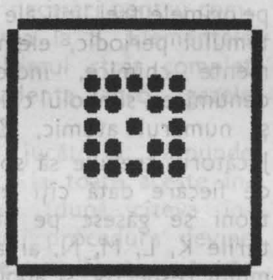
```
 900 RETURN
1000 INK 5
1010 FOR c=cmin TO cmax
1020 PRINT AT lmin-1,c;"<CAPS 3>"
1025 PRINT AT lmax+1,c;"<3>"
1030 NEXT c
```

```
1040 FOR l=lmin TO lmax
1050 PRINT AT l,cmin-1;"<5>"
1060 PRINT AT l,cmax+1;"<CAPS 5>"
1070 NEXT l
1080 INK 2
1100 RETURN
```

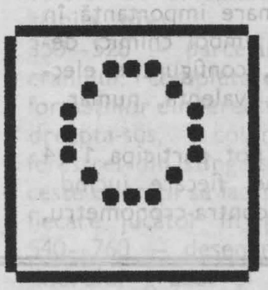
generatia numarul 1



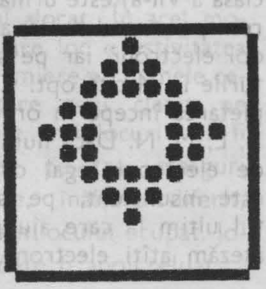
generatia numarul 1



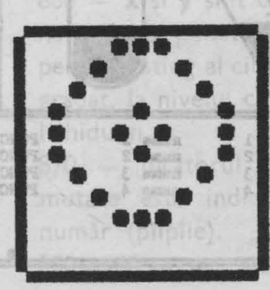
generatia numarul 2



generatia numarul 3



generatia numarul 4



VALENȚA

Programul face parte dintr-o serie de jocuri pe calculator, utile în învățarea chimiei în gimnaziu, realizate de prof. Anișoara Păun și Gh. Păun. Noțiunile de chimie implicate sînt destul de numeroase și de mare importanță în școală: simbol chimic, denumire, configurație electronică, valență, număr atomic.

La joc pot participa 1—4 persoane, fiecare jucînd solitar, contra-cronometru, pentru a acumula cît mai multe puncte. Înainte de partidă, jucătorii trebuie să-și spună numele (8 caractere, completat cu spații la dreapta dacă este cazul), apoi trebuie să aleagă nivelul de joc (între 1 = lent, 2 = mediu, 3 = rapid). După începerea jocului propriu-zis, ecranul va arăta ca în figura 18. Distingem « fereastra » din stînga-sus, pe care sînt desenate un cilindru gradat și un pahar Berzelius, «fereastra» din dreapta-sus în care se desfășoară dialogul cu calculatorul, și partea inferioară, unde sînt precizate numele jucătorilor și punctele realizate de fiecare.

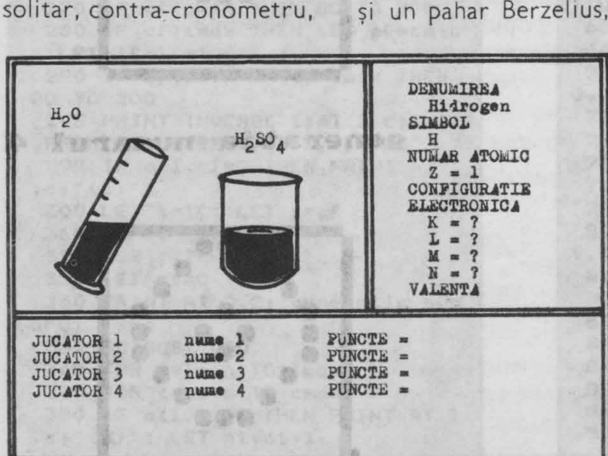


Fig. 18

Cilindrul gradat are rol de ceas. El se umple treptat cu lichid. În momentul în care conținutul cilindrului se revarsă în pahar, timpul afectat jucătorului respectiv s-a terminat (și aflăm atunci că în cilindru se găsea apă iar în pahar acid sulfuric; operația de turnare a apei în acid este interzisă, fapt sonorizat adecvat de program și « răsplătit » cu oprirea jocului). Viteza de umplere a cilindrului depinde de nivelul de joc; la orice eroare, nivelul apei în cilindru crește.

În partea dreaptă a ecranului, programul prezintă succesiv, alese la întîmplare de pe primele trei linii ale sistemului periodic, elemente chimice, indicînd denumirea, simbolul chimic și numărul atomic, Z.

Jucătorul trebuie să spună de fiecare dată cîți electroni se găsesc pe straturile K, L, M, N, ai atomului respectiv și apoi să calculeze valența.

Regula de completare a straturilor (se învață în clasa a VII-a) este următoarea: pe stratul K se așază doi electroni iar pe straturile L, M cîte opt. Completarea începe în ordinea K, L, M, N. Dacă numărul de electroni (egal cu Z) este insuficient, pe stratul ultim la care ajungem așezăm atîți electroni cîți

ne rămîn. De exemplu, fosforul are Z = 15; pe stratul K vom avea doi electroni, pe stratul L vom avea opt (ambele sînt complete), iar pe stratul M vor rămîne 15 — 2 — 8 = 5 electroni. La stratul N nici nu ajungem. Valența unui element este egală cu numărul de electroni pe care un atom trebuie să-i piardă sau să-i primească pentru a avea ultimul strat complet. Dacă pe ultimul strat se găsesc 1, 2, 3 electroni, este mai ușor să cedeze acești electroni decît să primească 7, 6, respectiv, 5 electroni și atunci valența va fi chiar egală cu numărul de electroni (la fel în cazul existenței a 4 electroni pe ultimul strat). Dacă pe ultimul strat se găsesc 5, 6, 7 electroni, atunci valența va fi 3, 2, respectiv, 1, adică numărul de electroni necesari pentru completarea la 8. Elementele cu ultimul strat complet au valența zero (gazele rare).

Dacă jucătorul răspunde corect la toate aceste întrebări (după cîteva încercări, procedura devine foarte simplă), el primește un număr de puncte egal cu Z.

Se continuă astfel pînă ce fiecare jucător își epuizează timpul alocat. În acel moment are loc « festivitatea de premiere »: numele celor care s-au clasat pe primele trei locuri vor fi înscrise pe trei steaguri albastre, înălțate diferit, conform locului ocupat. Jocul poate fi apoi reluat.

Deprinderea în joacă a modului de completare a configurației electronice și de calculare a valenței elementelor uzuale poate fi de mare ajutor în învățarea chimiei. Iar procedura nu este deloc complicată: programul a fost experimentat și cu copiii care nu știau chimie (clasa a V-a și VI-a) și s-a constatat că nu au avut nici o dificultate în a reține (desigur, fără să înțeleagă în detaliu despre ce este vorba) algoritmul respectiv (aplicîndu-l probabil fără referire la elementul concret propus de program, plecînd doar de la numărul atomic Z).

Descrierea programului

15 (GOSUB 9000) — pe liniile 9000 — 9010 se desenează cifra 2 ca indice (pentru H_2O, de exemplu), iar pe liniile 9020 — 9030 se desenează cifra 4 ca indice (vor fi identificate prin « s », respectiv, « r » în modul grafic).

20 (GOSUB 5000) — pe liniile 5000 — 5040 se desenează picătura care va cădea din cilindru la umplere (« a » în modul grafic), iar pe liniile 5050 — 5090 se desenează punctele care « decorează » fondul ferestrei din stînga-sus a ecranului (« b » în modul grafic); pe liniile 5100 — 5130 se desenează semnul minus care se scrie ca «exponent» la litera « e », indicînd un electron « e^- » (« c » în modul grafic).

25—30 — instrucțiunile de joc.

100 (GOSUB 7000) — completarea matricelor cu date: **d$** conține denumirile elementelor, **s$** conține simbolurile chimice, **z** conține numerele atomice, **v** conține valențele, **t** conține configurațiile electronice pentru fiecare element (numărul de electroni de pe fiecare strat K, L, M, N), **v$** conține valențele scrise cu cifre romane, conform uzanței din chimie; vectorul **p** va conține punctele realizate de fiecare jucător.

110—130 — începerea jocului.

140 — 195 — precizarea numărului de jucători (variabila **nj**).

200 — numele jucătorilor sînt reținute în **j$**.

210—280 — se introduc numele jucătorilor.

290—340 — precizarea nivelului de dificultate (variabila **niv**).

350—528 — parcelarea ecranului, completarea informațiilor din fereastra din dreapta-sus, « colorarea » ferestrei din stînga-sus (aceste operații se fac pentru fiecare jucător în parte).

540—760 — desenarea cilindrului gradat și a paharului Berzelius.

830—850 — se scriu numele jucătorilor.

860 — **x** și **y** sînt coordonatele unui punct situat pe peretele stîng al cilindrului gradat, la nivelul curent al lichidului.

870 — jucătorul aflat la mutare este indicat prin număr (pîlpîie).

880—900 — se alege la

întîmplare un element dintre cele 20.

905 — se șterg cîmpurile ocupate eventual cu informații despre un alt element (apărut mai devreme pe ecran).

910—930 — se scriu pe ecran denumirea, simbolul și numărul atomic al elementului ales.

940—1010 — se cere numărul de electroni de pe straturile K, L, M, N.

1000 — în caz de eroare, nivelul apei din cilindru crește.

1004 — se testează dacă cilindrul s-a umplut.

1020—1060 — se cere valența, se verifică, se penalizează eroarea, dacă este cazul.

1070—1160 — revărsarea cilindrului.

1170 — se indică conținutul cilindrului (H_2O) și al paharului Berzelius (H_2SO_4).

1180—1230 — sonorizarea încheierii rîndului jucătorului **j**.

1250 — rîndul lui **j** s-a încheiat.

1260 — răsplătirea jucătorului **j**, care continuă jocul (se reia de la linia 900 cu alegerea unui nou element).

1290—1390 — se desenează pe ecran trei steaguri la înălțimi diferite.

1400—1440 — se identifică și se scrie pe cel mai înalt steag numele cîștigătorului întrecerii.

1460—1500 — se identifică și se scrie pe steagul al doilea numele ocupantului locului doi.

1520—1560 — se scrie pe steagul al treilea numele ocupantului locului trei.

1560—1575 — sonorizarea sfîrșitului de partidă.

1580—1610 — opțiunea de reluare a programului.

```
  5 BORDER 1: PAPER 7: CLS
 10 PRINT AT 3,6; INVERSE 1;"
 V A L E N T A  "
 15 RESTORE 9000: GO SUB 9000
 20 RESTORE : GO SUB 5000
 25 PRINT AT 6,8;"Instructiuni"
 30 PRINT AT 8,2;"Cunoscind num
arul atomic, tre-buie sa deducet
i configuratia  electronica si
valenta unor ele-mente alese la
intimplare din-  tr-o lista anum
ita. La fiecare  raspuns corect
primiti cite  "; INVERSE 1;"Z";
INVERSE 0;"  puncte. Puteti juca
 pina la um- plerea cilindrului
gradat; dupa aceea..."
 100 GO SUB 7000
 110 PRINT AT 18,6; FLASH 1;"Inc
epem (d/n) ?"
 120 PAUSE 0: LET r$=INKEY$: BEE
P .02,12: BEEP .02,22
 130 IF r$<>"d" THEN GO TO 120
 140 CLS : PRINT AT 10,2;"Citi j
ucatori (1,2,3,4) ?"
 150 PAUSE 0: LET r$=INKEY$: BEE
P .02,12: BEEP .02,22
 160 IF r$>="1" AND r$<="4" THEN
 LET nj=VAL r$: GO TO 200
 195 BEEP 1,-6: GO TO 140
 200 DIM j$(4,8)
 210 FOR i=1 TO nj
 230 PRINT : PRINT "Jucator ";i;
" (8 litere): ";
 240 FOR j=1 TO 8
 250 PAUSE 0: LET j$(i,j)=INKEY$
: BEEP .02,12: BEEP .02,22
 260 PRINT j$(i,j);
 270 NEXT j
```

```
 280 NEXT i
 285 FOR i=1 TO 4: LET p(i)=0: N
EXT i
 290 PRINT : PRINT : PRINT "
Nivel ?"
 295 PRINT : PRINT "   (1=lent,
2=mediu, 3=rapid) "
 300 PAUSE 0: LET r$=INKEY$: BEE
P .02,12: BEEP .02,22
 310 IF r$>="1" AND r$<="3" THEN
 LET niv=VAL r$: GO TO 350
 340 BEEP 1,-6: GO TO 300
 350 PAUSE 40: CLS
 360 FOR j=1 TO nj
 365 INK 2
 370 PRINT AT 0,0;"<CAPS 8888888
88888888888888888888888888>"
 380 PRINT AT 16,0;"<CAPS 888888
888888888888888888888888888>"
 390 PRINT AT 21,0;"<CAPS 888888
888888888888888888888888888>"
 400 FOR i=1 TO 20
 410 PRINT AT i,0;"<CAPS 8>": PR
INT AT i,31;"<CAPS 8>"
 420 NEXT i
 430 FOR i=1 TO 15
 440 PRINT AT i,18;"<CAPS 8>"
 450 NEXT i
 455 INK 0
 460 PRINT AT 2,19; INVERSE 1;"D
ENUMIREA    "
 470 PRINT AT 4,19; INVERSE 1;"S
IMBOL       "
 480 PRINT AT 6,19; INVERSE 1;"N
UMAR ATOMIC"
 485 PRINT AT 7,22;"Z =   "
 490 PRINT AT 8,19; INVERSE 1;"C
ONFIGURATIE"
```

```
 500 PRINT AT 9,19; INVERSE 1;"E
LECTRONICA"
 510 PRINT AT 10,21;"K =      "
 512 PRINT AT 11,21;"L =      "
 514 PRINT AT 12,21;"M =      "
 516 PRINT AT 13,21;"N =      "
 520 PRINT AT 14,19; INVERSE 1;"
VALENTA       "
 523 FOR i=1 TO 15
 525 PRINT AT i,1;"<bbbbbbbbbbbb
bbbbb>"
 528 NEXT i
 540 PLOT 20,100: DRAW 50,50
 550 DRAW 12,-12,PI/3
 560 DRAW -12,12,PI/3
 570 PLOT 20,100: DRAW 12,-12,PI
/3: DRAW 50,50
 580 PLOT 23,103: DRAW -6,-6,PI:
 DRAW 12,-12,PI/3: DRAW 6,6,PI
 590 PLOT 20,100: DRAW 22,0
 591 PLOT 21,99: DRAW 21,0
 592 PLOT 22,98: DRAW 19,0
 593 PLOT 23,97: DRAW 17,0
 594 PLOT 23,96: DRAW 16,0
 595 PLOT 24,95: DRAW 13,0
 596 PLOT 25,94: DRAW 12,0
 597 PLOT 25,93: DRAW 12,0
 598 PLOT 26,92: DRAW 10,0
 599 PLOT 27,91: DRAW 8,0
 600 PLOT 28,90: DRAW 7,0
 601 PLOT 29,89: DRAW 5,0
 610 FOR i=6 TO 50 STEP 6
 620 PLOT 20+i,100+i: DRAW 3,-3
 630 NEXT i
 640 PLOT 60,105: DRAW 0,-51: DR
AW 34,0,PI/3: DRAW 0,45: DRAW 4,
4: DRAW -40,0,PI/3: DRAW 40,0,PI
/3
 650 FOR i=54 TO 74
 660 PLOT 60,i: DRAW 34,0
 670 NEXT i
 680 FOR i=1 TO 3
 690 PLOT 60+i,54-i: DRAW 34-2*i
,0
 700 NEXT i
 710 PLOT 65,50: DRAW 24,0
 720 PLOT 69,49: DRAW 17,0
 730 PLOT 60,75: DRAW 6,0: PLOT
60,76: DRAW 4,0: PLOT 60,77: DRA
W 2,0
 740 PLOT 94,75: DRAW -6,0: PLOT
 94,76: DRAW -4,0: PLOT 94,77: D
RAW -2,0
 750 PLOT 60,77: DRAW 34,0,-PI/3
 760 PLOT 68,77: DRAW 20,0,-PI/4
: DRAW -20,0,-PI/4
 830 FOR t=1 TO nj
 840 PRINT AT 16+t,1; INVERSE 1;
"JUCATOR ";t; INVERSE 0;j$(t); I
NVERSE 1;"PUNCTE:"; INVERSE 0;"
 ";p(t)
 850 NEXT t
 860 LET x=20: LET y=100
 870 PRINT AT 16+j,9; FLASH 1;j
```

```
 880 LET al=RND
 900 LET al=INT (RND*20)+1
 905 PRINT AT 3,21;"          ": P
RINT AT 5,21;"          ": PRINT
AT 7,21;"          ": PRINT AT 10,
24;"       ": PRINT AT 11,24;"
   ": PRINT AT 12,24;"       ": P
RINT AT 13,24;"        ": PRINT AT
 15,21;"          "
 910 PRINT AT 3,21;d$(al)
 920 PRINT AT 5,22;s$(al)
 930 PRINT AT 7,22;z(al)
 940 FOR i=1 TO 4
 950 LET te=t(al,i)
 960 IF te=0 THEN GO TO 1020
 965 PRINT AT 9+i,24; FLASH 1;"?
"
 970 IF INKEY$="" THEN PAUSE 60*
(4-niv)
 972 IF INKEY$="" THEN GO TO 100
0
 975 LET r$=INKEY$: BEEP .02,12:
 BEEP .02,22
 980 IF r$>="0" AND r$<="9" THEN
 LET r=VAL r$: GO TO 990
 985 BEEP .5,-6: GO TO 1000
 990 IF r=te AND te=1 THEN PRINT
 AT 9+i,24;" e";"<c>": GO TO 101
0
 995 IF r=te THEN PRINT AT 9+i,2
4;te;" e";"<c>": GO TO 1010
1000 LET x=x+1: LET y=y+1: PLOT
x,y: DRAW 23,0
1004 IF x<=58 THEN GO TO 970
1006 GO TO 1070
1010 NEXT i
1020 PRINT AT 15,22; FLASH 1;"?"
1030 IF INKEY$="" THEN PAUSE 60*
(4-niv)
1032 IF INKEY$="" THEN GO TO 106
0
1035 LET r$=INKEY$: BEEP .02,12:
 BEEP .02,22
1040 IF r$>="0" AND r$<="9" THEN
 LET r=VAL r$: GO TO 1050
1045 BEEP .5,-6: GO TO 1060
1050 IF r=v(al) THEN PRINT AT 15
,22;u$(al): GO TO 1260
1060 LET x=x+1: LET y=y+1: PLOT
x,y: DRAW 23,0: IF x<=58 THEN GO
 TO 1030
1070 FOR i=1 TO 6
1075 PAUSE 10
1080 FOR t=5 TO 11
1090 PRINT AT t,10; OVER 0;"<b>"
1100 PRINT AT t+1,10; OVER 1;"<a
>"
1110 NEXT t
1120 PRINT AT 12,10; OVER 1;"<b>
"
1130 FOR t=48 TO 50
1140 BEEP .0001*t,t
1150 NEXT t
1160 NEXT i
```

```
1170 PRINT AT 5,2; FLASH 1;"H";"
(s)";"O": PRINT AT 13,12; FLASH
1;"H";"(s)";"SO";"(r)"
1180 FOR i=1 TO 14
1190 FOR t=0 TO 7
1200 BORDER t
1210 BEEP .01,6*t
1220 NEXT t
1230 NEXT i
1240 PRINT AT 16+j,9;j
1250 GO TO 1280
1260 LET p(j)=p(j)+z(al): PRINT
AT 16+j,27;p(j)
1265 PAUSE 40
1270 GO TO 900
1280 NEXT j
1290 BORDER 3: PAPER 5: CLS : IN
K 1
1310 PLOT 13,0: DRAW 0,170: PLOT
 15,0: DRAW 0,170
1320 PLOT 93,0: DRAW 0,150: PLOT
 95,0: DRAW 0,150
1330 PLOT 173,0: DRAW 0,130: PLO
T 175,0: DRAW 0,130
1340 FOR i=0 TO 40: PLOT 16,i: D
RAW 68,0: NEXT i
1350 FOR i=0 TO 40: PLOT 96,i: D
RAW 68,0: NEXT i
1360 FOR i=0 TO 40: PLOT 176,i:
DRAW 68,0: NEXT i
1370 FOR i=0 TO 125: PLOT 16,i:
DRAW OVER 1,68,0: PLOT 16,i+40:
DRAW 68,0: NEXT i
1380 FOR i=0 TO 105: PLOT 96,i:
DRAW OVER 1,68,0: PLOT 96,i+40:
DRAW 68,0: NEXT i
1390 FOR i=0 TO 85: PLOT 176,i:
DRAW OVER 1,68,0: PLOT 176,i+40:
 DRAW 68,0: NEXT i
1400 LET max=-10
1410 FOR j=1 TO nj
1420 IF p(j)>max THEN LET max=p(
j): LET loc=j
1430 NEXT j
1440 PRINT AT 3,2;j$(loc): LET p
(loc)=-11
1450 IF nj=1 THEN GO TO 1580
1460 LET max=-10
1470 FOR j=1 TO nj
1480 IF p(j)>max THEN LET max=p(
j): LET loc=j
1490 NEXT j
1500 PRINT AT 6,12;j$(loc): LET
p(loc)=-11
1510 IF nj=2 THEN GO TO 1580
1520 LET max=-10
1530 FOR j=1 TO nj
1540 IF p(j)>max THEN LET max=p(
j): LET loc=j
1550 NEXT j
1560 PRINT AT 8,22;j$(loc)
1570 FOR i=-2 TO 4
1572 BEEP .05,12: BEEP .5,i*10
1575 NEXT i
1580 PRINT AT 20,6; INVERSE 1;"A
LT JOC (d/n) ?"
1590 PAUSE 0: LET r$=INKEY$: BEE
P .1,12: BEEP .1,22
1600 IF r$="d" THEN GO TO 140
1610 STOP
5000 FOR x=0 TO 7
5010 READ y
5020 POKE USR "a"+x,y
5030 NEXT x
5040 DATA BIN 00010000,BIN 00010
000,BIN 00111000,BIN 00111000,BI
N 01111100,BIN 01111100,BIN 0011
1000,0
5050 FOR x=0 TO 7
5060 READ y
5070 POKE USR "b"+x,y
5080 NEXT x
5090 DATA BIN 00100000,0,0,0,BIN
 00000010,0,0,0
5100 FOR x=0 TO 7
5110 READ y
5120 POKE USR "c"+x,y
5125 NEXT x
5130 DATA 0,BIN 00111110,0,0,0,0
,0,0
5140 RETURN
7000 DIM d$(20,8): DIM s$(20,2):
 DIM z(20): DIM t(20,4): DIM v(2
0): DIM p(4): DIM u$(20,3)
7010 FOR i=1 TO 20
7020 READ d$(i)
7030 NEXT i
7040 DATA "hidrogen","heliu","li
tiu","beriliu","bor","carbon","a
zot","oxigen","fluor","neon","so
diu","magneziu","aluminiu","sili
ciu","fosfor","sulf","clor","arg
on","potasiu","calciu"
7050 FOR i=1 TO 20
7060 READ s$(i)
7070 NEXT i
7080 DATA "H","He","Li","Be","B"
,"C","N","O","F","Ne","Na","Mg",
"Al","Si","P","S","Cl","Ar","K",
"Ca"
7090 FOR i=1 TO 20
7100 READ v(i)
7110 NEXT i
7120 DATA 1,0,1,2,3,4,3,2,1,0,1,
2,3,4,3,2,1,0,1,2
7130 FOR i=1 TO 20
7140 LET z(i)=i
7150 NEXT i
7170 FOR i=1 TO 20
7180 FOR j=1 TO 4
7190 READ t(i,j)
7200 NEXT j
7210 NEXT i
7220 DATA 1,0,0,0,2,0,0,0,2,1,0,
0,2,2,0,0,2,3,0,0,2,4,0,0,2,5,0,
0,2,6,0,0,2,7,0,0,2,8,0,0,2,8,1,
0,2,8,2,0,2,8,3,0,2,8,4,0,2,8,5,
0,2,8,6,0,2,8,7,0,2,8,8,0,2,8,8,
```

```
1,2,8,8,2
7230 FOR i=1 TO 20
7240 READ u$(i)
7250 NEXT i
7260 DATA "I","O","I","II","III"
,"IV","III","II","I","O","I","II
","III","IV","III","II","I","O",
"I","II"
7270 RETURN
9000 FOR i=0 TO 7: READ x: POKE
```

```
USR "s"+i,x: NEXT i
9010 DATA 0,0,0,BIN 00110000,BIN
 01001000,BIN 01010000,BIN 00100
000,BIN 01111000
9020 FOR i=0 TO 7: READ x: POKE
USR "r"+i,x: NEXT i
9030 DATA 0,0,0,BIN 00010000,BIN
 00110000,BIN 01010000,BIN 01111
000,BIN 00010000
9040 RETURN
```

Jocuri de reflexe

Cele mai răspîndite jocuri pe calculator sînt cele numite « de reflexe », prin care sînt puse la încercare (implicit, antrenate) îndemînarea, rapiditatea în evaluarea unor situații dinamice, în luarea unor decizii și exactitatea transpunerii acestora în practică. De obicei, jocurile de acest gen sînt bazate pe un « scenariu » care se vrea realist și spectaculos în același timp, însoțit, desigur, de o « scenografie » cît mai convingătoare. Este aproape imposibil de făcut o descriere cît de cît cuprinzătoare a domeniului. O clasificare superficială poate distinge între jocuri solitare, jocuri pur competitive (doi jucători manevrează pe ecran doi « luptători » de cele mai diferite tipuri), jocuri la care participă mai multe persoane, fiecare însă pe rînd (biliard, flippere etc.). O clasificare mai de substanță se poate încerca plecînd de la tema jocurilor. Unele se referă la conducerea unui automobil, elicopter, avion, avînd deci un pronunțat caracter de simulator (sînt probabil cele mai valoroase jocuri de reflexe), altele, spuneam, se referă la lupte de diferite feluri (arte marțiale,

pistoale etc.); o largă clasă se dezvoltă în jurul temei călătoriei, a labirintului (un personaj trebuie condus pe un itinerar unde îl pîndesc pericole, dar îl așteaptă și recompense); multe jocuri simulează o întrecere sportivă (solitar — tenis la perete, de exemplu — competitiv, contra calculatorului), altele au ca trăsătură principală duelul de artilerie (de obicei împotriva unor invadatori extratereștri. . .). Bineînțeles rămîn pe dinafară multe alte jocuri (inclusiv unele din lucrarea de față).

În general, realizarea unui joc de reflexe cere eforturi de programare deosebite (grafică, animație, sunete, evaluarea scorului), depășind adesea nivelul limbajului BASIC. Pot fi însă realizate jocuri reușite și în acest cadru și sperăm că programele care urmează vor demonstra aceasta, contribuind, în același timp, la «ascuțirea reflexelor» cititorului, incitîndu-l la realizarea unor jocuri mai elaborate.

ping - pong

Pe mijlocul ecranului apare o «paletă» de tenis de masă așezată vertical. Ea poate fi mutată în sus apăsînd tasta **S** și în jos apăsînd tasta **J**. De pe marginea de jos a ecranului, dintr-o poziție aleasă la întîmplare de program, pleacă o «minge de tenis», spre stînga sau spre dreapta (direcția este aleasă tot la întîmplare), pe o dreaptă care face un unghi de 45° cu orizontala. Mingea ricoșează de la orice perete întîlnit (tot la 45°) și își continuă mișcarea pînă ce face 50 de schimbări de direcție (atunci jocul se încheie) sau întîlnește paleta. La fiecare întîlnire cu paleta, se acumulează un punct și mingea pleacă din nou de pe marginea de jos

a ecranului. Punctele realizate și numărul de ciocniri de marginile ecranului sînt indicate în fiecare moment pe rîndul imediat inferior spațiului de joc (numărul ciocnirilor se cumulează, la relansarea mingii de pe linia de jos nu se pleacă de la zero, ci de la numărul ciocnirilor anterioare).
După încheierea unui joc (după 50 de ciocniri ale mingii de margine), programul întreabă « Alt joc (d/n)? », putînd fi reluat (apăsînd tasta D) sau oprit.
Pentru a acumula cît mai multe puncte, în afară de manevrarea rapidă și exactă a paletei, ceea ce presupune o bună coordonare a mișcăriilor, de mare importanță este anticiparea direcției de deplasare a mingii pentru a o intercepta la locul potrivit.
O posibilă **modificare** a programului este mărirea paletei; în program ea ocupă două caractere pe verticală, dar ar putea fi desenată din trei-patru caractere pentru a ușura sarcina jucătorului. Eventual, acesta poate fi un parametru, descriind nivelul de dificultate pentru care optează jucătorul. Schimbările necesare în program rămîn în seama cititorului.

Descrierea programului

10 — **p** este numărul de puncte realizat de jucător la un moment dat; **t** este « timpul » de joc alocat (numărul de ciocniri de marginile ecranului).
100—120 — paleta este desenată pe liniile **loc, loc+1,** pe coloana 15.
130 — **x** și **y** sînt coordonatele curente ale mingiei.
140 — **dir** este direcția de deplasare a mingiei (între 1 și 8, conform celor patru direcții și două sensuri posibile pe fiecare direcție).
160—170 — se verifică atingerea marginilor.
180—190 — comenzile jucătorului.
210—220 — mingea își continuă drumul.
230—240 — se verifică atingerea paletei.
280—310 — se deplasează paleta în sus.
320—350 — se deplasează paleta în jos.
390—470 — la atingerea unui perete se scade o unitate din **t** (pînă devine zero) și se schimbă direcția de mișcare a mingii, modificînd variabila **dir**.
480—540 — la efectuarea a 50 de lovituri se întreabă dacă se reia jocul.
1000—1020 — matricea **d** conține regulile de mișcare a mingiei (pe primele două coloane) și de schimbare a direcției (pe ultima coloană), în funcție de direcția anterioară (variabila **dir**).
1030—1040 — se desenează paleta (« a » în modul grafic).
1050—1060 — se desenează mingea (« b » în modul grafic) (« a » grafic apare pe liniile 120, 290, 300, 330, 340 iar « b » grafic apare pe liniile 150, 200, 220).

```
  5 GO SUB 1000
 10 PAPER 6: BORDER 1: CLS : LE
T p=0: LET t=50
 80 FOR i=0 TO 31: PRINT AT 20,
i; INK 1;"(CAPS 8)": NEXT i
 90 PRINT AT 21,0; INVERSE 1;"
 PUNCTE =  ";p;"    RESURSE =
30 "
 100 LET loc=9
 120 PRINT AT loc,15;"(a)": PRIN
T AT loc+1,15;"(a)"
 130 LET x=19: LET y=INT (RND*28
)+2
 140 LET dir=4*(INT (RND*2)+1)
 150 PRINT AT x,y;"(b)"
 160 IF x=0 OR x=19 THEN GO TO 3
60
 170 IF y=0 OR y=31 THEN GO TO 3
60
 180 LET r$=INKEY$: IF r$="s" TH
EN GO TO 280
 190 IF r$="j" THEN GO TO 320
 200 PRINT OVER 1;AT x,y;"(b)"
 210 LET x=x+d(dir,1)
 211 LET y=y+d(dir,2)
 220 PRINT OVER 1;AT x,y;"(b)"
 230 IF y<>15 THEN GO TO 160
 240 IF x<>loc AND x<>loc+1 THEN
 GO TO 160
 250 FOR i=1 TO 6: BEEP .1,i*i:
NEXT i
 260 LET p=p+1: PRINT AT 21,12;p
 270 PRINT AT loc,15;" ": PRINT
AT loc+1,15;" ": GO TO 110
 280 IF loc=0 THEN GO TO 200
 290 PRINT OVER 1;AT loc+1,15;"(
a)"
 300 PRINT OVER 1;AT loc-1,15;"(
a)"
 310 LET loc=loc-1: GO TO 200
 320 IF loc=18 THEN GO TO 200
 330 PRINT OVER 1;AT loc,15;"(a)
"
 340 PRINT OVER 1;AT loc+2,15;"(
a)"
 350 LET loc=loc+1: GO TO 200
 360 LET t=t-1: BEEP .03,-3: BEE
P .003,4: NEXT i
 370 PRINT AT 21,29;"  ": PRINT
AT 21,29;t
 380 IF t=0 THEN GO TO 480
 390 IF x>0 THEN GO TO 415
 395 IF y=0 THEN LET dir=8: GO T
O 200
 400 IF y=31 THEN LET dir=4: GO
TO 200
 405 IF dir=1 THEN LET dir=7: GO
TO 200
 410 IF dir=5 THEN LET dir=3: GO
TO 200
 415 IF x<19 THEN GO TO 470
 420 IF y=0 THEN LET dir=2: GO T
O 200
 430 IF y=31 THEN LET dir=6: GO
TO 200
 450 IF dir=7 THEN LET dir=1: GO
TO 200
 460 IF dir=3 THEN LET dir=5: GO
TO 200
 470 LET dir=d(dir,3): GO TO 200
 480 FOR j=1 TO 3: FOR i=7 TO 1
STEP -1
 490 BORDER i: BEEP .05,i*j: BEE
P .02,j*j
 495 LET r$=INKEY$
 500 NEXT i: NEXT j
 510 PRINT AT 6,8;"Alt joc (d/n)
?"
 515 PAUSE 10: LET r$=INKEY$
 520 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 530 IF r$="d" THEN GO TO 10
 540 STOP
1000 DIM d(8,3)
1010 FOR i=1 TO 8: FOR j=1 TO 3:
 READ d(i,j): NEXT j: NEXT i
1020 DATA -1,-1,2,-1,1,3,1,1,4,1
,-1,1,-1,1,6,-1,-1,7,1,-1,8,1,1,
5
1030 FOR i=0 TO 7: READ x: POKE
USR "a"+i,x: NEXT i
1040 DATA BIN 01111100,BIN 01010
100,BIN 01010100,BIN 01111100,BI
N 01111100,BIN 01010100,BIN 0101
0100,BIN 01111100
1050 FOR i=0 TO 7: READ x: POKE
USR "b"+i,x: NEXT i
1060 DATA BIN 00011000,BIN 00101
100,BIN 01101110,BIN 01011110,BI
N 01101110,BIN 00101100,BIN 0001
1000,0
1070 RETURN
```

cursa cu obstacole

cursa cu obstacole - cursa cu obstacole

Un joc care se poate juca cu numai o singură tastă și aceasta poate fi oricare de pe tastatură (în afară de cele două taste de control CAPS SHIFT și SYMBOL SHIFT).

La începutul jocului se solicită introducerea vitezei alergătorului (care reprezintă de fapt gradul de dificultate al jocului), un număr de la 1 la 10. Apoi, pe ecran apare parcursul cu cele 12 obstacole așezate cîte 3 pe 4 linii drepte, distanța dintre obstacole fiind egală, indicîndu-se locul de plecare și cel de sosire.

În partea de jos a ecranului (sub parcurs) apare un text care indică permanent numărul de obstacole dărîmate și viteza alergătorului. Se dă un semnal sonor și alergătorul pornește din partea stîngă cu o viteză proporțională cu valoarea introdusă de jucător în prealabil. Acționînd o tastă la momentul potrivit, alergătorul va sări peste obstacole. Dar dacă tasta se va acționa prea devreme sau prea tîrziu, obstacolul va fi dărîmat. Cursa terminată, jocul se poate relua (eventual de alt jucător cu comanda RUN). Pentru ca jocul să se reia automat se poate introduce linia **235 RUN**.

Modificare sugerată: o cursă cu obstacole cu doi alergători, fiecare condus de către un jucător.

Descrierea programului

20 (GO SUB 1200) – apelarea subrutinei de introducere a vitezei alergătorului și de validare a acesteia (un număr cuprins între 1 și 10). Pentru viteză se folosește variabila **v**. În linia 1230 a subrutinei se atribuie variabilei **d** (exprimă durata de parcurgere a distanței dintre doi « pași ») valoarea **11-v**. De fapt, valoarea **d** reprezintă cît timp « stă » alergătorul pînă ce face următorul « pas », cînd va fi șters de pe vechea poziție și desenat pe următoarea.

30 – inițializarea variabilelor: **no** reprezintă numărul de obstacole dărîmate; **n** reprezintă numărul de « pași » efectuați după trecerea ultimului obstacol; **t** specifică dacă s-a acționat prea devreme o tastă (1 = da). La începutul jocului toate aceste variabile au valoarea 0.

40, 55, 57 – tablou de afișaj.

50 (GO SUB 1000) – apelarea subrutinei de definire a caracterelor grafice corespunzătoare alergătorului (1000–1060) și a traseului cu obstacole (1100–1135). Obstacolele sînt aranjate pe 4 linii drepte care corespund liniilor 3, 5, 7 și 9 ale ecranului grafic (vezi linia de program 1100 în care variabila **l** reprezintă linia).

Liniile de program 1105 și 1120 servesc la desenarea obstacolelor. Se observă

din linia 1105 că pe fiecare din cele 4 linii obstacolele sînt așezate în alt mod (și anume pe coloana ecranului grafic **12—l**). Din linia 1120 se poate remarca distanța egală dintre obstacole și anume de 9 celule caracter. Caracterul grafic pentru obstacol nedărîmat se obține acționînd tasta 5 (cu CAPS SHIFT) în modul grafic.
65 — semnalul sonor de start.
70—210—ciclu de parcurs al celor 4 linii drepte.
80—210 — ciclu de parcurs al unei linii (este o linie a ecranului grafic de 32 de celule — poziții caracter —, de la 0 la 31). Alergătorul se deplasează din poziție în poziție, dar pentru a se sugera mișcarea, pe pozițiile corespunzătoare coloanelor cu numere pare, alergătorul va avea alt aspect față de alergătorul afișat pe pozițiile corespunzătoare coloanelor cu numere impare.
Din această cauză, în linia 80 apare STEP 2: caracterul alergător se afișează pe poziția corespunzătoare liniei **l** și coloanei **c**, dar din două în două coloane (pe coloanele cu număr par).
90—afișarea alergătorului pe poziția corespunzătoare liniei **l** și coloanei **c** (cu număr par). Caracterul « alergător » (de pe coloană cu număr par) se obține prin acționarea tastei **Q** în modul grafic.
100 — pauză care servește la reglarea vitezei. Dacă s-a introdus inițial o viteză **v = 1**, atunci pauza între doi « pași » va fi **d = 11 — v = 10**, adică circa o cincime de secundă (PAUSE 10). După pauză se va șterge caracterul alergătorului de pe poziție pentru afișarea lui pe coloana următoare.
110 — variabila **k** indică dacă alergătorul are de gînd să « sară » sau nu (**k = 0** nu sare — la începutul jocului —, **k = 1** sare). Numărul de « pași » **n** efectuați după ultimul obstacol crește cu 2.
120 — dacă s-a apăsat prea devreme pe o tastă.
130 — alergătorul a ajuns la obstacol (dacă **n = 10** înseamnă că a parcurs 10 « pași » de la ultimul obstacol). În acest caz, dacă se acționează o tastă, saltul va avea loc. În acest scop numărul liniei **l** va scădea cu o unitate (caracterul « alergător » urmează să se afișeze pe o linie mai sus, deasupra obstacolului), iar **k** se va face **1**, indicație că alergătorul va « sări ». Dacă nu se acționează nici o tastă se va trece la linia următoare (140).
140 — variabila **t** devine **O** și imediat este readusă la **1** dacă s-a acționat o tastă înainte de a se fi ajuns la un obstacol.
150 — pauză pentru a se trece la poziția următoare și se afișează caracterul « alergător » (de pe coloană cu număr impar) pe poziția următoare. Acest caracter se obține prin acționarea tastei **P** în modul grafic.
160 — apelarea subrutinei de dărîmare a obstacolului (GO SUB 1300). La dărîmarea obstacolului numărul de « pași » după trecerea de ultimul obstacol **n** devine **O** (linia 1300), iar numărul de obstacole dărîmate **no** crește cu **1** (linia 1310). Acest număr (scorul) se afișează imediat după căderea obstacolului (linia 1350). În locul obstacolului « în picioare » se afișează unul dărîmat care se obține acționînd tasta **3 (cu CAPS SHIFT)** în modul grafic (linia 1340). Dărîmarea obstacolului este însoțită de un sunet (linia 1320).
170 — dacă alergătorul nu sare, **(k = 0)** și nu se găsește deasupra unui obstacol, atunci se șterge de pe poziția respectivă.
180 — dacă alergătorul sare, atunci se șterge de pe poziția respectivă și se face pregătirea pentru « coborîrea » lui pe linia inferioară (numărul liniei grafice **l** crește cu o unitate).
230, 240 — sfîrșitul jocului.

```
 5 BORDER 6: PAPER 7: INK 9
10 PRINT "Cursa cu obstacole"
20 GO SUB 1200
30 LET no=0: LET n=0: LET t=0
40 PRINT AT 12,1;0;" obstacol
 darimat  viteza ";v
```

```
50 GO SUB 1000
55 PRINT AT 2,0;"PLECARE"
57 PRINT AT 10,24;"SOSIRE"
65 BEEP 1,5
70 FOR l=3 TO 9 STEP 2
80 FOR c=0 TO 31 STEP 2
```

```
  90 PRINT AT l,c;"<q>"
 100 PAUSE d: PRINT AT l,c;" "
 110 LET k=0: LET n=n+2
 120 IF t=1 THEN GO TO 140
 130 IF n=10 THEN IF INKEY$<>""
THEN LET l=l-1: LET k=1
 140 LET t=0: IF n<>10 AND INKEY
$<>"" THEN LET t=1
 150 PAUSE d: PRINT AT l,c+1;"<p
>"
 160 IF n=10 THEN GO SUB 1300
 170 IF k=0 AND n<>0 THEN PAUSE
d: PRINT AT l,c+1;" "
 180 IF k=1 THEN PAUSE d: PRINT
AT l,c+1;" ";: LET l=l+1
 200 NEXT c
 210 NEXT l
 230 BEEP .2,13: BEEP .2,16
 240 STOP
 000 DATA 8,28,61,90,24,164,66,1
 010 DATA 16,56,16,58,84,48,40,6
3
1020 FOR s=80 TO 81
1030 FOR i=0 TO 7
1040 READ a: POKE USR CHR$ s+i,a
```

RICOȘEU

Pe ecran apar o serie de obstacole (« pereți ») orizontale și verticale, de diferite dimensiuni, cinci ținte și o steluță, așezată pe linia de jos a spațiului de joc. Steluța poate fi deplasată spre stînga apăsînd tasta 5, spre dreapta cu ajutorul tastei 8 și poate fi lansată — apăsînd tasta 4 sau tasta 9 — pe o direcție care face un unghi de 45° cu orizontala, înspre stînga, respectiv, înspre dreapta. Dacă

```
1050 NEXT i
1060 NEXT s
1100 FOR l=3 TO 9 STEP 2
1105 PRINT AT l,12-l;
1110 FOR h=1 TO 3
1120 PRINT "<CAPS 5>      ";
1130 NEXT h
1135 NEXT l
1150 RETURN
1200 BEEP .2,20
1210 INPUT "introduceti viteza(
1 si 10) ";v
1220 IF v<1 OR v>10 THEN GO TO 1
200
1230 LET d=11-v
1240 RETURN
1300 LET n=0
1310 IF k=1 THEN RETURN
1320 BEEP .1,30
1330 LET no=no+1
1340 PRINT AT l,c+1;"<CAPS 3>"
1350 PRINT AT 12,1;no
1360 IF no=2 THEN PRINT AT 12,11
;"e darimate"
1380 RETURN
```

în drumul ei, steluța întîlnește un perete, ea ricoșează (tot la 45° față de perete) și își continuă drumul pînă ce face 40 de asemenea schimbări de direcție. După 40 de ciocniri steluța revine pe linia de jos și poate fi din nou controlată (așezată în poziție dorită și lansată în direcția potrivită).

Scopul fiecărei încercări este lovirea celor cinci ținte (direct sau în urma unui număr arbitrar de ricoșeuri). Pot fi făcute opt încercări, după care jocul se încheie (cu întrebarea « Alt joc (d/n)? » — deci poate fi reluat).

Modificare interesantă: Pentru că, după lansarea steluței, programul « joacă» singur ar putea fi atractivă posibilitatea deplasării unui perete/unor pereți, eventual introducerea în partea de jos a spațiului de joc a unui perete manevrabil, pentru a putea modifica traiectoria steluței.

Descrierea programului

10 (GOSUB 1000) — matricea **m** conține modificările de coordonate ale steluței, în funcție de direcția de mișcare, iar matricea **d** conține schimbările direcției de mișcare la întîlnirea obstacolelor.
20—40 — se colorează ecranul.

70–299 – se desenează obstacolele; cele verticale sînt obținute scriind cifra 1 (cu aceeași culoare ca fondul) iar cele orizontale scriind cifra 2.

300 – se plasează țintele.

330 – **inc** este numărul de încercări (pînă la 8), **sc** este scorul realizat.

340–350 – steluța are coordonatele **x**, **y**.

370–400 – comenzile de mișcare a steluței pe linia de jos și de lansare a ei.

420–440 – deplasarea steluței spre stînga.

450–470 – deplasarea steluței spre dreapta.

500–520 – noile coordonate ale steluței.

525–550 – testare a cîmpului curent (țintă? obstacol? liber?).

560–590 – deplasare în spațiu liber.

600–650 – lovirea unei ținte.

660–740 – opțiunea de reluare a jocului.

720 – lovirea unui obstacol.

730 – se schimbă direcția de deplasare (în funcție de direcția anterioară și de sensul de deplasare pe direcție, indicat de variabila **p**, fixată la liniile 540, 550).

740–780 – s-au făcut deja 40 de schimbări de direcție; steluța revine pe linia de jos a ecranului (dacă nu s-au făcut încă 8 încercări).

```
 10 BORDER 1: GO SUB 1000
 20 FOR i=0 TO 21: FOR j=0 TO 2
8 STEP 4
 30 PRINT AT i,j; INK 4;"<CAPS
8888>": BEEP .02,i+j
 40 NEXT j: NEXT i
 50 DIM a$(22): LET a$="  SUPER
 BILIARD"
 60 FOR i=1 TO 22: PRINT AT i-1
,0; INVERSE 1;a$(i): NEXT i
 70 INK 0: PAPER 0
 80 FOR i=1 TO 31
 90 PRINT AT 0,i;"1": PRINT AT
20,i;"1"
100 NEXT i
110 FOR i=7 TO 25
120 PRINT AT 5,i;"1"
130 NEXT i
140 PRINT AT 10,15;"111"
142 PRINT AT 16,15;"111"
150 FOR i=1 TO 21
160 PRINT AT i,1;"2"
165 PRINT AT i,31;"2"
180 NEXT i
190 FOR i=8 TO 16
200 PRINT AT i,8;"2"
202 PRINT AT i,4;"2"
205 PRINT AT i,24;"2"
207 PRINT AT i,28;"2"
220 NEXT i
230 FOR i=11 TO 15
240 PRINT AT i,13;"2"
245 PRINT AT i,19;"2"
260 NEXT i
290 PAPER 4
295 PRINT AT 11,4;" ": PRINT AT
11,28;" "
297 PRINT AT 12,4;" ": PRINT AT
12,28;" "
299 INVERSE 1
300 PRINT AT 3,11;"#": PRINT AT
3,16;"#": PRINT AT 3,21;"#"
305 PRINT AT 8,16;"#"
310 PRINT AT 13,16;"#"
320 PRINT AT 21,1;"NUMAR INCERC
ARI:": PRINT AT 21,20;"LOVITURI:
"
330 INVERSE 0: LET inc=0: LET s
c=0
340 PRINT AT 19,16; FLASH 1;"*"
: BEEP .1,22
350 LET x=19: LET y=16
360 PAUSE 0: LET r$=INKEY$: BEE
P .1,22
370 IF r$="5" THEN GO TO 420
380 IF r$="8" THEN GO TO 450
390 IF r$="4" THEN LET dir=4: G
O TO 500
400 IF r$="9" THEN LET dir=3: G
O TO 500
410 BEEP 1,-6: GO TO 360
420 IF y=2 THEN GO TO 360
430 PRINT AT x,y;" ": LET y=y-1
440 PRINT AT x,y;"*": GO TO 360
450 IF y=30 THEN GO TO 360
460 PRINT AT x,y;" ": LET y=y+1
470 PRINT AT x,y;"*": GO TO 360
500 LET inc=inc+1: PRINT AT 21,
18;inc: LET lov=0
510 LET x1=x+m(dir,1)
520 LET y1=y+m(dir,2)
525 IF SCREEN$ (x1,y1)=" " THEN
 GO TO 560
530 IF SCREEN$ (x1,y1)="#" THEN
 GO TO 600
540 IF SCREEN$ (x1,y1)="1" THEN
 LET p=1: GO TO 720
550 IF SCREEN$ (x1,y1)="2" THEN
 LET p=2: GO TO 720
560 PRINT AT x,y;" "
570 LET x=x1: LET y=y1
580 PRINT AT x,y;"*"
590 GO TO 510
600 FOR i=1 TO 5: FOR j=7 TO 1
610 BORDER j: BEEP .02,i*j: BEE
P .01,i*j+10
```

```
620 NEXT j: NEXT i
630 PRINT AT x,y;" "
640 PRINT AT x1,y1;"*": LET sc=
sc+1
650 IF sc<5 THEN GO TO 510
660 PRINT AT 18,2;"TERMINAT"
670 FOR i=1 TO 10: LET a=INT (R
ND*20)
680 BEEP .1,a: NEXT i
690 PRINT AT 19,2;"Alt joc (d/n
) ?"
700 PAUSE 0: IF INKEY$<>"d" THE
N STOP
710 BEEP .3,22: GO TO 20
720 LET lov=lov+1: PRINT AT 21,
29;lov: BEEP .01,33
730 IF lov<40 THEN LET dir=d(di
r,p): GO TO 510
740 PRINT AT 21,29; FLASH 1;lov
750 FOR i=1 TO 10: LET a=INT (R
ND*10)
760 BEEP .02,a: NEXT i: PAUSE 3
0
770 IF inc=8 THEN GO TO 660
775 PRINT AT x,y;" "
780 PRINT AT 21,28;" 0 ": GO TO
340
1000 DIM m(4,2): DIM d(4,2)
1010 FOR i=1 TO 4: FOR j=1 TO 2:
READ m(i,j): NEXT j: NEXT i
1015>DATA 1,-1,1,1,-1,1,-1,-1
1020 FOR i=1 TO 4: FOR j=1 TO 2:
READ d(i,j): NEXT j: NEXT i
1040 DATA 4,2,3,1,2,4,1,3
1050 RETURN
2700 PRINT AT 5,7;"3": PRINT AT
5,25;"3"
2750 PRINT AT 10,13;"3": PRINT A
T 10,19;"3"
```

DEPLASARE

Pe ecran apar, așezate la întîmplare, 30 de semne &, iar în centru o steluță. Aceasta poate fi deplasată în sus sau în jos, apăsînd tastele A, respectiv, Z, și la stînga sau la dreapta, apăsînd tastele N, respectiv, M. În mișcarea ei, steluța « marchează» drumul parcurs. Problema care se pune este colectarea de cît mai multe semne & (trecînd peste ele), dar fără a depăși marginea spațiului de joc și fără a reveni cu steluța într-un cîmp în care ea a mai fost o dată (fără a intersecta drumul parcurs).

Programul permite patru niveluri de dificultate (1 = lent, 2 = mediu, 3 = rapid, 4 = expert), ca timp permis pentru deplasare; sînt alocate 100 de unități de timp, de mărimi diferite, corespunzătoare nivelului. Contorul de măsurare a timpului scurs nu înaintează atunci cînd se apasă una dintre tastele care comandă deplasarea steluței, dar, deplasînd fără rost steluța se « consumă » din spațiul disponibil pe ecran, deci șansa de a intersecta drumul parcurs crește. La încheierea unui joc, programul poate fi reluat.

Descrierea programului

5 — **sc** este scorul curent; **ti** este timpul scurs (crește pînă la 100, cînd jocul se oprește); GOSUB 1000: după instrucțiuni (linia 1000) se cere nivelul de dificultate dorit (liniile 1020 — 1040), apoi se marchează marginile spațiului de joc (linia 1050) și se distribuie la întîmplare 30 de trofee pe ecran (linia 1070); nivelul de dificultate este indicat de variabila **niv**.

10 — coordonatele curente ale steluței sînt **a** și **b**.

15 — « ceasul», funcție de nivelul ales.

20—50 — comenzile de deplasare a steluței (coordonatele viitoare se modifică pe loc).

70 — se verifică dacă steluța a atins marginea sau intersectează propriul drum.

80 — se verifică dacă se culege un trofeu.

100 — deplasarea steluței într-un loc gol.

120—160 — la atingerea marginii, intersectarea drumului sau cînd **ti** = **100** (linia 15), jocul se oprește, putînd fi reluat (liniile 140 — 150).

```
5 CLS : BORDER 5: LET sc=0: LET ti=0: GO SUB 1000
10 LET a=10: LET b=15: PRINT AT a,b; FLASH 1;"*"
15 PAUSE 6*(4.3-niv): IF INKEY$="" THEN LET ti=ti+1: PRINT AT 21,29;ti: BEEP .05,33: IF ti=100 THEN GO TO 120
20 IF INKEY$="z" THEN LET a=a+1: PRINT PAPER 4;AT a-1,b;"*": GO TO 70
30 IF INKEY$="a" THEN LET a=a-1: PRINT PAPER 4;AT a+1,b;"*": GO TO 70
40 IF INKEY$="m" THEN LET b=b+1: PRINT PAPER 4;AT a,b-1;"*": GO TO 70
50 IF INKEY$="n" THEN LET b=b-1: PRINT PAPER 4;AT a,b+1;"*": GO TO 70
60 GO TO 15
70 IF SCREEN$ (a,b)="*" THEN BEEP .1,22: BEEP .2,33: GO TO 120
80 IF SCREEN$ (a,b)="&" THEN LET sc=sc+1: BEEP .1,40
90 PRINT AT 21,13;sc
100 PRINT AT a,b; INK 2; FLASH 1;"*"
110 GO TO 15
120 BEEP 1,1: CLS : PRINT AT 6,8; FLASH 1;"  TERMINAT !  "
130 PRINT AT 10,7;"Scorul = ";sc;" puncte"
140 PRINT AT 15,7;"Alt joc (d/n) ?"
150 PAUSE 0: IF INKEY$="d" THEN GO TO 5
160 STOP
1000 CLS : PRINT AT 4,10; INVERSE 1;"DRUM BUN !": PRINT : PRINT : PRINT "Trebuie sa culegeti cit mai mul-te trofee "; INVERSE 1;"&"; INVERSE 0;", deplasind pe ecran "; INVERSE 1;"*"; INVERSE 0;", fara sa atingeti marginea si fara sa va intersectati drumul. Timpul este limitat (la 100).": PRINT : PRINT "Comenzi: A= in sus,Z= in jos,N= la stinga,M= la dreapta"
1020 PRINT AT 18,1;"NIVEL": PRINT " 1=lent,2=mediu,3=rapid,4=expert"
1025 PAUSE 0: LET q$=INKEY$: BEEP .1,40
1030 IF q$>="1" AND q$<="4" THEN LET niv=VAL q$: GO TO 1050
1040 BEEP 1,-6: GO TO 1020
1050 CLS : FOR i=0 TO 21: PRINT INK 2; PAPER 2;"*";AT i,31;"*": NEXT i: PRINT INK 2; PAPER 2;AT 0,0;"********************************";AT 20,0;"********************************"
1060 PRINT AT 21,1; INK 1; INVERSE 1;" PUNCTE =    0       TIMP =    0 "
1070 FOR i=0 TO 30: LET x=INT (RND*19)+1: LET y=INT (RND*30)+1: PRINT INK 1;AT x,y;"&": NEXT i
1080 RETURN
```

cărare

Pe ecran apare o linie șerpuită (o «cărare»), de culoare închisă, care merge din colțul din stînga-jos pînă în cel din dreapta-sus, unde se află o steluță pîlpîitoare. Tot din stînga-jos pleacă o linie subțire de culoare contrastînd cu culoarea fondului pe care ea se mișcă. Ea se deplasează inițial în sus, dar poate fi făcută să cotească spre stînga, apăsînd tasta C și spre dreapta, apăsînd tasta V. Atenție: virarea **la stînga** și **la dreapta** se raportează la direcția curentă de mers. De exemplu, dacă linia este în mișcare în sus, atunci indicația stînga-dreapta corespunde înțelesului uzual, dar dacă linia se deplasează spre dreapta, orizontal, atunci « spre stînga » în raport cu direcția de mișcare este în

sus, iar « spre drepta » este în jos. Obiectivul este conducerea liniei subțiri de-a lungul « cărării », pînă la steaua din colțul din dreapta-sus al ecranului. Orice pas în afara « cărării » este numărat, iar la efectuarea a 100 asemenea pași eronați, jocul se încheie (și poate fi reluat din starea inițială, deoarece programul întreabă în final « Alt joc (d/n) ? »). De reținut că apăsarea tastelor trebuie să se facă cu finețe, altfel, zăbovind asupra lui C, de exemplu, programul va interpreta aceasta ca o dublă comandă « spre stînga » (sau chiar triplă), ceea ce revine la întoarcerea în sensul contrar de deplasare. Mai mult, reparcurgerea unei porțiuni de drum (pe cărare) va fi interpretată ca eroare și pașii respectivi vor fi contorizați ca atare (nu este însă interzisă parcurgerea unei porțiuni de cărare de mai multe ori, dar pe linii distincte).

Modificare posibilă: introducerea și a unei comenzi de oprire, pentru momentele de derută, dar cu adăugarea unei limite de timp (eventual și a unor niveluri diferite de joc), înainte de care trebuie ajuns la țintă.

Descrierea programului

5—15 — matricea **a** conține modificările coordonatelor punctului în mișcare, în funcție de direcția de mișcare anterioară și de comenzile jucătorului.
21—41 — desenarea cărării de pe ecran și a steluței din colțul din dreapta-sus.
50 — coordonatele curente ale punctului care se mișcă sînt **x** și **y**.
55 — **ab** este numărul de abateri (pași ai punctului, efectuați în afara cărării).
70—80 — comenzile de modificare a direcției de mers.
90—95 — coordonatele viitoare ale punctului mobil.
100 — punctul nu are voie să iasă din ecran.
105 — un pas în afara cărării.
120—180 — terminarea jocului (fie au fost efectuați 100 de pași în afara cărării, fie a fost părăsit spațiul de joc).
200 — verificarea atingerii stelei-țintă.
230—260 — schimbarea direcției de deplasare, spre stînga, respectiv, spre dreapta.

```
  5 DIM a(4,2)
  10 FOR i=1 TO 4: READ a(i,1):
READ a(i,2): NEXT i
  15 DATA 0,1,-1,0,0,-1,1,0
  20 CLS : BORDER 1: PAPER 6
  21 PRINT
  22 PRINT " <CAPS 888888>  <CAP
S 8888888888888888888888> "
  23 PRINT " <CAPS 8>     <CAPS 8
>  <CAPS 8>                    <
CAPS 8> "
  24 PRINT " <CAPS 8>     <CAPS 8
>  <CAPS 8>  <CAPS 888888888888>
  <CAPS 88>"; FLASH 1;"*"; FLASH
 0;" <CAPS 8> "
  25 PRINT " <CAPS 888> <CAPS 88
>  <CAPS 8>  <CAPS 8>          <
CAPS 8> <CAPS 88>   <CAPS 8> "
  26 PRINT "    <CAPS 8> <CAPS 8>
  <CAPS 8> <CAPS 8>            <
CAPS 8> <CAPS 8>     <CAPS 8> "
  27 PRINT "    <CAPS 8> <CAPS 8>
  <CAPS 88> <CAPS 888888888>  <
CAPS 8> <CAPS 8888> <CAPS 8> "
  28 PRINT "    <CAPS 8> <CAPS 8>
  <CAPS 8>              <CAPS 8> <
CAPS 8>     <CAPS 8> <CAPS 8> "
  29 PRINT "    <CAPS 8> <CAPS 8>
  <CAPS 8>              <CAPS 8> <
CAPS 888888> <CAPS 8> "
  30 PRINT "    <CAPS 8> <CAPS 8>
  <CAPS 8>              <CAPS 8>
     <CAPS 8> "
  31 PRINT "    <CAPS 8> <CAPS 8>
  <CAPS 8>  <CAPS 888888>    <CAP
S 8>              <CAPS 8> "
  32 PRINT "    <CAPS 8> <CAPS 8>
  <CAPS 8>  <CAPS 8>      <CAPS 8>
   <CAPS 888888888> <CAPS 8> "
  33 PRINT " <CAPS 888> <CAPS 8>
  <CAPS 8>  <CAPS 8>      <CAPS 8>
             <CAPS 8> <CAPS 8> "
  34 PRINT " <CAPS 8>    <CAPS 8>
  <CAPS 8>  <CAPS 888>   <CAPS 8>
             <CAPS 8> <CAPS 8> "
  35 PRINT " <CAPS 888> <CAPS 8>
  <CAPS 8>     <CAPS 8>  <CAPS 8>
    <CAPS 88888>  <CAPS 8> <CAPS
 8> "
  36 PRINT "    <CAPS 8> <CAPS 8>
  <CAPS 8>     <CAPS 8>  <CAPS 8>
    <CAPS 8>    <CAPS 8>  <CAPS 8
 > <CAPS 8> "
  37 PRINT "    <CAPS 8> <CAPS 8>
  <CAPS 8>     <CAPS 8>  <CAPS 8>
    <CAPS 8>    <CAPS 8>  <CAPS 8
 > <CAPS 8> "
  38 PRINT "    <CAPS 8> <CAPS 8>
```

```
 <CAPS 8888> <CAPS 8> <CAPS 88
8888> <CAPS 8888> <CAPS 8> "
  39 PRINT " <CAPS 8> <CAPS 8>
      <CAPS 888>              <
CAPS 8> "
  40 PRINT " <CAPS 8> <CAPS 8>
                 <CAPS 8>
"
  41 PRINT " <CAPS 8> <CAPS 88
8888888888888888888888888> "
  50 LET x=27: LET y=13
  55 LET ab=0: PRINT AT 21,0;ab
  60 PLOT x,y: LET dir=1: BEEP .
2,30
  70 IF INKEY$="c" THEN GO TO 23
0
  80 IF INKEY$="v" THEN GO TO 25
0
  90 LET x=x+a(dir,1)
  95 LET y=y+a(dir,2)
 100 IF x=0 OR x=255 OR y=0 OR y
=175 THEN GO TO 120
 105 IF POINT (x,y)=0 THEN LET a
b=ab+1: BEEP .02,-6: PRINT AT 21
,0;ab
 110 IF ab<100 THEN GO TO 190
 120 FOR i=1 TO 5
 130 FOR j=7 TO 1 STEP -1: BORDE
R j: BEEP .02,i*j: NEXT j
 140 NEXT i
 150 PRINT AT 21,0;"Alt joc (d/n
) ?                  "
 155 PAUSE 20
 160 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 170 IF r$="n" THEN STOP
 180 GO TO 20
 190 PLOT OVER 1;x,y: BEEP .003,
33
 200 IF ABS (y-148)>4 OR ABS (x-
228)>4 THEN GO TO 70
 210 PRINT AT 21,0;"BRAVO  BRAVO
  BRAVO !!"
 220 GO TO 120
 230 LET dir=dir+1: IF dir=5 THE
N LET dir=1
 240 GO TO 90
 250 LET dir=dir-1: IF dir=0 THE
N LET dir=4
 260 GO TO 90
```

STELE

Pe ecran apar, dispuse la întîmplare, 20 de steluțe, iar pe marginile de sus și de jos sînt marcate două liniuțe. Cea de sus poate fi deplasată spre stînga sau spre dreapta apăsînd tastele **T**, respectiv, **Y**, iar cea de jos poate fi deplasată în aceleași direcții apăsînd tastele **V**, respectiv, **B**. În orice moment, dacă se apasă tasta **F**, între cele două liniuțe se trasează o dreaptă. Toate stelele care sînt atinse de această dreaptă « se sting » (sînt șterse de pe ecran). Problema care se pune este « stingerea » tuturor stelelor printr-un număr cît mai mic de încercări. Evident, este avantajos ca dreptele să fie trasate în așa fel încît să fie atinse în același timp (coliniare sau aproape coliniare) cît mai multe stele. Pot fi făcute oricît de multe încercări. La epuizarea stelelor, jocul poate fi reluat.

Modificare posibilă: introducerea unei restricții de timp (eventual cu niveluri diferite de dificultate), pentru a mări dinamismul jocului.

Descrierea programului

20—80 — pregătirea ecranului
90 — coordonatele vîrfului liniuței de sus sînt **x1, y1.**
100 — coordonatele vîrfului liniuței de jos sînt **x2, y2.**
110 — vectorii **x, y** conțin coordonatele steluțelor.
120 — **lin** este numărul de linii trasate, iar **st** este numărul de steluțe existente pe ecran.
130—160 — se precizează la întîmplare 30 de coordonate.

170—250 — se desenează pe ecran 20 de steluțe diferite, verificînd, deci, dacă există stele care coincid — linia 190; variabila **k** indică numărul de coordonate disponibile pentru alegerea celor 20 de perechi distincte (inițial — linia **120 — k** era egal cu 30).

230 — dacă nu găsim 20 de steluțe distincte, atunci programul se reia de la început.

290—300 — se desenează liniuțele de pe margini.

350—390 — comenzile jucătorului: mișcarea liniuței de sus (liniile 350—360), a celei de jos (liniile 370—380) sau trasarea unei drepte între acestea (linia 390).

410—440 — deplasarea spre stînga a liniuței de sus.

450—480 — deplasarea spre dreapta a liniuței de sus.

490—520 — deplasarea spre stînga a liniuței de jos.

530—560 — deplasarea spre dreapta a liniuței de jos.

570—585 — trasarea unei drepte între liniuțe și numărarea ei.

590—660 — verificarea dacă au fost atinse stele (ștergem stelele pe rînd — linia 610 — și verificăm dacă rămîne loc gol în urma lor — linia 620 —, caz în care linia n-a trecut pe acolo, deci refacem steluța).

680—710 — la terminarea stelelor (**st = 0** la linia 650), se comentează acest lucru și se cere opțiunea de reluare.

```
  10 CLS : OVER 0: BORDER 1
  20 FOR i=0 TO 31
  30 PRINT AT 0,i;"<CAPS 8>"
  40 PRINT AT 20,i;"<CAPS 8>"
  50 NEXT i
  60 FOR i=1 TO 19
  70 PRINT AT i,0;"<CAPS 8>"
  75 PRINT AT i,31;"<CAPS 8>"
  80 NEXT i
  90 LET x1=128: LET y1=16
 100 LET x2=128: LET y2=167
 110 DIM x(30): DIM y(30)
 120 LET lin=0: LET st=20: LET k
=30
 130 FOR i=1 TO 30
 140 LET x(i)=INT (RND*17)+2
 150 LET y(i)=INT (RND*25)+3
 160 NEXT i
 165 PRINT AT x(1),y(1);"*": BEE
P .01,22
 170 FOR i=2 TO 20
 180 FOR j=1 TO i-1
 190 IF x(i)=x(j) AND y(i)=y(j)
THEN GO TO 220
 200 NEXT j
 210 PRINT AT x(i),y(i);"*": BEE
P .01,x(i)+y(i): GO TO 250
 220 LET x(i)=x(k): LET y(i)=y(k
)
 230 LET k=k-1: IF k<20 THEN GO
TO 10
 240 GO TO 180
 250 NEXT i
 260 PRINT AT 21,0; INVERSE 1;"
Linii =       Stele ramase =    "
 270 PRINT AT 21,10;lin;AT 21,29
;st
 280 OVER 1
 290 PLOT x1,y1: DRAW 0,-4
```

```
 300 PLOT x2,y2: DRAW 0,4
 340 BEEP .001,25
 350 IF INKEY$="v" THEN GO TO 41
0
 360 IF INKEY$="b" THEN GO TO 45
0
 370 IF INKEY$="t" THEN GO TO 49
0
 380 IF INKEY$="y" THEN GO TO 53
0
 390 IF INKEY$="f" THEN GO TO 57
0
 400 GO TO 340
 410 IF x1=10 THEN GO TO 340
 420 PLOT x1,y1: DRAW 0,-4
 430 LET x1=x1-2: PLOT x1,y1: DR
AW 0,-4
 440 GO TO 340
 450 IF x1=246 THEN GO TO 340
 460 PLOT x1,y1: DRAW 0,-4
 470 LET x1=x1+2: PLOT x1,y1: DR
AW 0,-4
 480 GO TO 340
 490 IF x2=10 THEN GO TO 340
 500 PLOT x2,y2: DRAW 0,4
 510 LET x2=x2-2: PLOT x2,y2: DR
AW 0,4
 520 GO TO 340
 530 IF x2=246 THEN GO TO 340
 540 PLOT x2,y2: DRAW 0,4
 550 LET x2=x2+2: PLOT x2,y2: DR
AW 0,4
 560 GO TO 340
 570 FOR i=1 TO 10: BEEP .01,3*i
: NEXT i
 580 PLOT x1,y1: DRAW x2-x1,y2-y
1
 585 LET lin=lin+1: PRINT AT 21,
10; OVER 0;lin
```

```
590 FOR i=1 TO 20
600 IF x(i)=0 THEN GO TO 660
610 PRINT AT x(i),y(i);"*"
620 IF SCREEN$ (x(i),y(i))=" " THEN PRINT AT x(i),y(i);"*": GO TO 660
630 LET x(i)=0: LET st=st-1
640 PRINT AT 21,29; OVER 0;"  ";AT 21,29;st
650 IF st=0 THEN GO TO 680
660 NEXT i
670 GO TO 340
680 OVER 0: PRINT AT 19,1;"Ai reusit - felicitari !!!"
690 FOR i=1 TO 20: BEEP .01,INT (RND*30): NEXT i
700 PRINT AT 20,1;"Alt joc (d/n) ?        "
710 PAUSE 10: PAUSE 0: IF INKEY$="d" THEN GO TO 10
```

DOMINOURI CĂZĂTOARE

Pe ecran este « decupat » un spațiu de culoare deschisă (galben), care are la bază un « zid » orizontal de pătrate iar în centru-sus o steluță. Din această steluță vor începe să cadă dominouri, « cărămizi » formate din cîte două pătrate alipite. Aceste dominouri cad mai repede sau mai încet, în funcție de nivelul de joc ales la început; acesta poate fi 1 = lent, 2 = mediu, 3 = rapid, 4 = expert. Dominourile pot fi abătute spre stînga sau spre dreapta, apăsînd tastele B, respectiv, N și pot fi rotite în aer, spre stînga folosind tasta V și spre dreapta folosind tasta M. Rotirea este relativă la poziția anterioară: un domino vertical devine orizontal după rotire și invers. Fiecare rotire înseamnă și căderea cu un pas.

Jocul poate avea două obiective: (1) așezarea cît mai compactă a dominourilor pentru a avea pe ecran cît mai multe piese în momentul în care « construcția » atinge steluța din partea de sus a spațiului de joc (atunci partida se încheie) și (2) realizarea unor construcții cît mai ingenioase din dominourile respective. Desigur, primul obiectiv este interesant la niveluri superioare de dificultate, al doilea poate

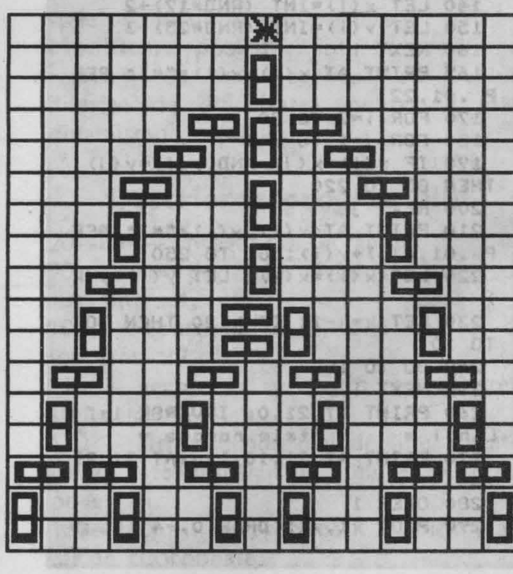


Fig.19

fi atractiv și la niveluri inferioare. Eventual, cele două probleme pot fi combinate: realizarea contracronometru a unei construcții date. O asemenea construcție este indicată în figura 19. La ce nivel de viteză de cădere a dominourilor puteți s-o realizați? În loc de dominouri, jocul ar putea folosi și altfel de piese. În particular, am putea face să cadă pentaminouri de forme precizate sau de oricare dintre cele 12 tipuri posibile (alese la întîmplare de program). Jocul (și programul) se complică însă atunci cînd piesele au forme diferite.

Descrierea programului

20—50 — se marchează pe ecran spațiul de joc.
60—70 — se desenează un pătrat (jumătate de domino); el va fi identificat de «a» în modul grafic.
80 — **p** este numărul de piese intrate în joc (scorul).
85 — se colorează baza spațiului de joc.
100—140 — precizarea nivelului de dificultate (variabila **niv**).
145 — se desenează șirul de dominouri de la bază și steluța.
150 — **x** și **y** sînt coordonatele dominoului curent (ale pătratului stîng sau de sus, în funcție de poziție, orizontală sau verticală).
160 — **dir** indică poziția — orizontală sau verticală — a dominoului.
170 — se desenează dominoul.
190 — «ceasul».
210—240 — comenzile jucătorului.
260—280 — dominoul cade liber, vertical **(dir = 1)**.
290 — se testează dacă s-a completat construcția pînă la steluță.
300—320 — jocul s-a terminat.
330—350 — opțiunea de reluare.
360—380 — dominoul cade liber, orizontal **(dir = 2)**.
390—440 — dominoul, în poziție verticală este deplasat cu o coloană spre stînga și cade cu un pas.
450—480 — dominoul, în poziție orizontală, este deplasat cu o coloană spre stînga și cade cu un pas.
490—530 — dominoul, în poziție verticală, este deplasat cu o coloană spre dreapta și cade cu un pas.
540—560 — dominoul, în poziție orizontală, este deplasat cu o coloană spre dreapta și cade cu un pas.
570—600 — dominoul, în poziție orizontală, este rotit spre stînga.
610—640 — dominoul, în poziție verticală, este rotit spre stînga.
650—690 — dominoul, în poziție orizontală, este rotit spre dreapta.
700—730 — dominoul, în poziție verticală, este rotit spre dreapta; — înainte de fiecare rotire se verifică dacă se poate continua căderea și dacă se poate efectua rotirea, iar după rotire se modifică adecvat variabila **dir.**

```
  10 BORDER 1: PAPER 6: CLS : IN
K 1
  20 FOR i=0 TO 21
  30 PRINT AT i,0; PAPER 3; INK
3;"********"
  40 PRINT AT i,23; PAPER 3; INK
 3;"*********"
  50 NEXT i: RESTORE
  60 FOR i=0 TO 7: READ x: POKE
USR "a"+i,x: NEXT i
  70 DATA 255,129,129,129,129,12
9,129,255
  80 LET p=0
  85 FOR i=19 TO 21: PRINT AT i,
8; INK 3;"<CAPS 8888888888888S>
": NEXT i
  90 PRINT AT 21,0; PAPER 0; INK
 7;"puncte = 0  "
 100 PRINT AT 6,8;"Nivel (1 - 4)
 ?"
```

```
 110 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 120 IF r$>="1" AND r$<="4" THEN
 LET niv=VAL r$: GO TO 140
 130 BEEP 1,-6: GO TO 110
 140 PRINT AT 6,8;"
  "
 145 PRINT AT 18,8; PAPER 2;"<aa
aaaaaaaaaaaa>": PRINT AT 0,15;
PAPER 0; INK 7;"*"
 150 LET x=1: LET y=15: LET p=p+
1
 160 LET dir=1: PRINT AT 21,9; P
APER 0; INK 7;p
 170 PRINT AT x,y; PAPER 4;"<a>"
: PRINT AT x+1,y; PAPER 4;"<a>"
 180 IF niv=4 THEN GO TO 200
 190 PAUSE (3.1-niv)*6
 210 IF INKEY$="b" THEN GO TO 39
0
```

```
220 IF INKEY$="n" THEN GO TO 49
0
230 IF INKEY$="v" THEN GO TO 57
0
240 IF INKEY$="m" THEN GO TO 65
0
250 IF dir=2 THEN GO TO 360
260 IF SCREEN$ (x+2,y)<>" " THE
N GO TO 290
270 PRINT AT x,y;" ": LET x=x+1
280 PRINT AT x+1,y; PAPER 4;"(a
)": BEEP .001,33: GO TO 180
290 IF x>1 THEN BEEP .001,33: G
O TO 150
300 FOR i=1 TO 5: FOR j=7 TO 1
STEP -1
310 BORDER j: BEEP .02,i*j+20
320 NEXT j: NEXT i
330 PRINT AT 15,2; PAPER 0; INK
7;"TERMINAT - Alt joc (d/n) ?"
340 PAUSE 10: PAUSE 0: IF INKEY
$="d" THEN GO TO 10
350 STOP
360 IF SCREEN$ (x+1,y)<>" " THE
N GO TO 290
365 IF SCREEN$ (x+1,y+1)<>" " T
HEN GO TO 290
370 PRINT AT x,y;"  ": LET x=x+
1
380 PRINT AT x,y; PAPER 4;"(aa)
": BEEP .001,33: GO TO 180
390 IF dir=2 THEN GO TO 450
400 IF SCREEN$ (x+2,y)<>" " TH
N GO TO 290
410 IF SCREEN$ (x+1,y-1)<>" " T
HEN GO TO 270
415 IF SCREEN$ (x+2,y-1)<>" " T
HEN GO TO 270
420 PRINT AT x,y;" ": PRINT AT
x+1,y;" "
430 LET x=x+1: LET y=y-1
440 PRINT AT x,y; PAPER 4;"(a)"
: PRINT AT x+1,y; PAPER 4;"(a)":
BEEP .001,33: GO TO 180
450 IF SCREEN$ (x+1,y)<>" " THE
N GO TO 290
455 IF SCREEN$ (x+1,y+1)<>" " T
HEN GO TO 290
460 IF SCREEN$ (x+1,y-1)<>" " T
HEN GO TO 370
470 PRINT AT x,y;"  ": LET x=x+
1: LET y=y-1
480 PRINT AT x,y; PAPER 4;"(aa)
": BEEP .001,33: GO TO 180
```

```
490 IF dir=2 THEN GO TO 540
500 IF SCREEN$ (x+2,y)<>" " THE
N GO TO 290
510 IF SCREEN$ (x+1,y+1)<>" " T
HEN GO TO 270
515 IF SCREEN$ (x+2,y+1)<>" " T
HEN GO TO 270
520 PRINT AT x,y;" ": PRINT AT
x+1,y;" "
530 LET x=x+1: LET y=y+1: GO TO
440
540 IF SCREEN$ (x+1,y)<>" " THE
N GO TO 290
545 IF SCREEN$ (x+1,y+1)<>" " T
HEN GO TO 290
550 IF SCREEN$ (x+1,y+2)<>" " T
HEN GO TO 370
560 PRINT AT x,y;"  ": LET x=x+
1: LET y=y+1: GO TO 480
570 IF dir=1 THEN GO TO 610
580 IF SCREEN$ (x+1,y)<>" " THE
N GO TO 290
585 IF SCREEN$ (x+1,y+1)<>" " T
HEN GO TO 290
590 LET dir=1: PRINT AT x,y+1;"
"
600 PRINT AT x+1,y; PAPER 4;"(a
)": BEEP .001,33: GO TO 180
610 IF SCREEN$ (x+2,y)<>" " THE
N GO TO 290
620 IF SCREEN$ (x+1,y-1)<>" " T
HEN GO TO 270
630 LET dir=2: PRINT AT x,y;" "
640 LET x=x+1: LET y=y-1: PRINT
AT x,y; PAPER 4;"(a)": BEEP .00
1,33: GO TO 180
650 IF dir=1 THEN GO TO 700
660 IF SCREEN$ (x+1,y)<>" " THE
N GO TO 290
665 IF SCREEN$ (x+1,y+1)<>" " T
HEN GO TO 290
680 LET dir=1: PRINT AT x,y;" "
690 LET y=y+1: PRINT AT x+1,y;
PAPER 4;"(a)": BEEP .001,33: GO
TO 180
700 IF SCREEN$ (x+2,y)<>" " THE
N GO TO 290
710 IF SCREEN$ (x+1,y+1)<>" " T
HEN GO TO 270
720 LET dir=2: PRINT AT x,y;" "
730 LET x=x+1: PRINT AT x,y; PA
PER 4;"(aa)": BEEP .001,33: GO T
O 180
```

Traversarea străzii

După ce se introduce nivelul de dificultate (4 niveluri posibile), apare pe ecran o arteră cu două sensuri de circulație, fiecare sens avînd cîte 3 benzi. Pe cele două sensuri circulă în mod continuu mașini: pe sensul din partea stîngă apar în mod aleator mașini (pe cele 3 benzi) care circulă de sus în jos (descendentă), iar pe sensul din partea dreaptă circulația este de jos în sus (ascendentă). Viteza de deplasare a mașinilor va fi în funcție de nivelul de dificultate ales. Pietonul (apare în partea stîngă a ecranului) poate traversa strada de la stînga la dreapta pe locul marcat. În acest scop el poate fi deplasat de jucător cu ajutorul **tastei 8 (pas înainte)**. De asemenea, jucătorul poate acționa și **tasta 5** pentru **mers înapoi,** dacă observă o mașină venind și nu poate evita altfel accidentul.

Orice impact al pietonului cu o mașină va fi marcat, dar (din fericire) pietonul va putea continua imediat traversarea. În partea de jos a ecranului se afișează în permanență numărul de accidente și numărul de traversări. Jocul are și un scop educativ, arătînd ce ușor se pot face accidente în caz de neatenție pe o arteră cu o circulație intensă. De asemenea, copiii vor înțelege (chiar și intuitiv) că cel mai periculos lucru pe o arteră de circulație este graba: într-adevăr, cu cît jucătorul se grăbește mai tare cu atît pericolul de accident este mai mare; în același timp jucătorul care nu se grăbește va obține un scor mai bun (mai puține accidente la același număr de traversări). Din această cauză nu s-a introdus în joc parametrul timp. Tot din același motiv nu s-a introdus în joc o limită a numărului de traversări: jocul poate dura oricît, el putînd fi însă oricînd întrerupt cu BREAK (tastele SPACE și CAPS SHIFT).

Modificări propuse: pornind de la această idee, jocul se poate complica și transforma într-un joc didactic pentru învățarea și respectarea regulilor de circulație. Pietonul se va putea deplasa și sus-jos și traversa strada pe locurile marcate. Se pot introduce și semafoare, în final obținîndu-se un punctaj în funcție de corectitudinea traversărilor și (eventual) a timpului în care se realizează.

Descrierea programului

22 — alegerea unuia dintre cele 4 niveluri de dificultate (variabila **dif**). Mașinile vor apărea și vor dispărea cu atît mai aproape de trecerea de pietoni (dînd astfel senzația unei viteze sporite din cauza unei frecvențe mai ridicate a lor în dreptul trecerii de pietoni) cu cît nivelul de dificultate introdus a fost mai mare.
25 — refuzul nivelurilor de dificultate fără interes.
27 — determinarea punctului de apariție a mașinilor care circulă de sus în jos (variabila **start 1**) și a mașinilor care circulă de jos în sus (variabila **start 2**).
28 — determinarea punctelor de dispariție a mașinilor (variabilele **fin 1** și **fin 2**). Valorile acestor variabile sînt calculate în

funcție de valoarea nivelului de dificultate ales (**dif**).

30 — rezervarea unui spațiu de memorie pentru matricea **b** ale cărei componente vor reprezenta pozițiile pe verticală (linia) ale tuturor mașinilor care circulă pe arteră (sînt 6 mașini). Aceste mașini sînt numerotate prin poziția lor pe orizontală: 6,7 și 8 pentru mașinile descendente, 13,14 și 15 pentru cele ascendente.

40 — inițializarea variabilelor care reprezintă numărul de traversări (**nt**) și respectiv numărul de accidente (**na**). La început acestea sînt egale cu **0.**

45 — inițializarea poziției (pe orizontală) a pietonului de fapt a poziției-coloană a caracterului pieton (variabila **c**). Caracterul « pieton » pornește de pe coloana **1** (deci din stînga ecranului).

55—65 — buclă de determinare a punctului de apariție a fiecărei mașini descendente (ales întîmplător dintr-un interval), mașinile pe fiecare bandă nemergînd în rînd (au puncte de apariție diferite).

70—90 — același lucru pentru mașinile ascendente.

1000(GOSUB 500) — apelarea subrutinei pentru desenarea arterei de circulație.

110 (GOSUB 600) — apelarea subrutinei de definire a caracterelor grafice corespunzătoare pentru pieton (două modele, liniile 600 și 610) și pentru mașină (două modele: mașină descendentă, linia 700, și mașină ascendentă, linia 690). În ciclurile 620—660 și 710—750 se citesc și se introduc în memorie valorile pentru definirea caracterelor pentru pietoni și respectiv mașini. Cele două caractere grafice pentru pieton se obțin cu **tasta Q** și respectiv **P** în modul grafic.

120 — **a** este in indicator al caracterului « pieton ». Poate fi **1** sau **0.**

140 — **d** este un indicator al deplasării pietonului: **0** = oprire, **1** = pas înainte, **—1** = pas înapoi. Inițial pietonul este oprit·

150 — dacă este acționată **tasta 5** pietonul face un **pas înapoi.**

160 — dacă este acționată **tasta 8** pietonul face un **pas înainte.**

170 — **acc** este un indicator pentru accident. El poate fi **0** (nu este accident) **1** (accident, dacă pietonul se află sub o mașină sau se va afla sub o mașină), sau **2** (accident dacă pietonul era și se va găsi sub mașină) (vezi linia 185).

175 — dacă pietonul nu se află sub o mașină, atunci caracterul « pieton » se șterge de pe poziția respectivă.

180 — se calculează noua poziție pe orizontală a pietonului (noua coloană **c**)

185 — se recalculează **acc.**

190 — dacă nu este accident, (**acc** = **0**) se afișează caracterul pieton pe noua poziție calculată.

195 — este accident **acc** (<>**0**): se emite un semnal sonor, valoarea lui **na** (numărul accidentelor) crește cu o unitate și se afișează această valoare în partea de jos a ecranului.

210—270 — buclă pentru deplasarea mașinilor descendente.

220 — ștergerea caracterului grafic corespunzător unei mașini.

230 — apariția unei noi mașini (în partea superioară a ecranului) de fiecare dată cînd o alta a dispărut (în partea de jos a ecranului).

240 — noua poziție pe verticală a unei mașini descendente (numărul liniei crește cu o unitate).

250 — afișarea pe această poziție a unui caracter grafic corespunzător unei mașini descendente. Caracterul grafic se va obține prin acționarea tastei **S** în modul grafic.

310—370 — buclă pentru deplasarea mașinilor ascendente.

340 — noua poziție pe verticală a unei mașini ascendente (numărul liniei scade cu o unitate).

350 — afișarea pe această poziție a unui caracter grafic corespunzător unei mașini, ascendente. Caracterul grafic se va obține prin acționarea tastei **R** în modul grafic.

400 — sfîrșitul unei traversări: recalcularea numărului de traversări efectuate și afișarea sa, pietonul dispare și se reinițializează poziția sa de pornire (de data aceasta de la coloana 2).

410 — se reia ciclul unei traversări.

```
 10 BORDER 6: PAPER 7: INK 0
 20 PRINT "Traversare"
 22 BEEP 0.2,20: INPUT "grad de
dificultate (1-4) ";dif
 25 LET dif=INT (dif): IF dif<1
OR dif>4 THEN GO TO 22
 27 LET start1=2+INT (dif/2): L
ET start2=16-INT (dif/2)
 28 LET fin1=16-dif: LET fin2=3
+dif
 30 DIM b(15)
 40 LET nt=0: LET na=0
 45 LET c=1
 50 PRINT AT 19,2;"accidente"
 52 PRINT AT 19,18;"traversari"
 55 FOR i=6 TO 8
 60 LET b(i)=INT (RND*4)+2
 65 NEXT i
 70 FOR i=13 TO 15
 80 LET b(i)=INT (RND*4)+16
 90 NEXT i
100 GO SUB 500
110 GO SUB 600
120 LET a=1
140 LET d=0
150 IF INKEY$="5" THEN LET d=-1
: LET a=NOT a
160 IF INKEY$="8" THEN LET d=1:
 LET a=NOT a
170 LET acc=POINT (c*8+4,88)
175 IF acc=0 THEN PRINT AT 10,c
;" "
180 LET c=c+d
185 LET acc=acc+POINT (c*8+4,88
)
190 IF acc=0 THEN PRINT AT 10,c
;CHR$ (159+a): GO TO 210
195 BEEP 0.2,15: LET na=na+1: P
RINT AT 20,2;na
210 FOR i=6 TO 8
220 PRINT AT b(i),i;" "
230 IF b(i)=fin1 THEN LET b(i)=
INT (RND*4)+start1
240 LET b(i)=b(i)+1
250 PRINT AT b(i),i;"<s>"
270 NEXT i
310 FOR i=13 TO 15
320 PRINT AT b(i),i;" "
330 IF b(i)=fin2 THEN LET b(i)=
INT (RND*4)+start2
340 LET b(i)=b(i)-1
350 PRINT AT b(i),i;"<r>"
370 NEXT i
400 IF c=18 THEN LET nt=nt+1: P
RINT AT 20,20;nt: PRINT AT 10,18
;" ": LET c=2
410 GO TO 140
500 PLOT 40,32: DRAW 0,56
510 PLOT 40,96: DRAW 0,56
520 PLOT 80,32: DRAW 0,56
530 PLOT 80,96: DRAW 0,56
540 PLOT 96,32: DRAW 0,56
550 PLOT 96,96: DRAW 0,56
560 PLOT 136,32: DRAW 0,56
570 PLOT 136,96: DRAW 0,56
580 RETURN
600 DATA 8,28,61,90,24,164,66,1
610 DATA 16,56,16,58,84,48,40,6
8
620 FOR s=80 TO 81
630 FOR i=0 TO 7
640 READ a
650 POKE USR CHR$ s+i,a
660 NEXT i: NEXT s
690 DATA 60,189,255,189,60,189,
255,189
700 DATA 189,255,189,60,189,255
,189,60
710 FOR s=82 TO 83
720 FOR i=0 TO 7
730 READ a
740 POKE USR CHR$ s+i,a
750 NEXT i: NEXT s
760 RETURN
```

Labirint

Pe ecran apare un labirint, dar nu se pune problema de a găsi o ieşire din labirint, ci de a reuşi să se conducă personajul «bun», astfel încît să scape de creaturile «rele». Se stabileşte iniţial numărul «răilor» (adversarilor) prin introducerea răspunsurilor la întrebarea pusă de calculator. Pot fi unul, doi sau trei «răi». Apoi, începe cursa, care este contracronometru. Jucătorul poate deplasa personajul «bun» orizontal sau vertical cu tastele: **5** — pentru stînga, **6** — pentru jos, **7** — pentru sus şi **8** — pentru dreapta. «Răii» cunosc poziţia «bunului», dar se deplasează într-un mod puţin dezordonat, mai ales atunci cînd se lovesc de pereţii labirintului. Ei pot avansa o poziţie sau chiar două, pe orizontală, dar şi pe diagonală, putînd «sări» peste pereţii subţiri. De aceea nu este indicat să staţi prea mult ascuns după pereţi, mai cu seamă că numărul de paşi pe care i-aţi făcut contează, fiind afişat la sfîrşitul jocului, fie că aţi fost «mîncat» de «răi», fie că aţi reuşit să scăpaţi pînă la sfîrşitul timpului regulamentar. În partea dreaptă jos a ecranului este afişat în permanenţă cronometrul care marchează timpul.

Modificări propuse. Se poate încerca şi varianta clasică a labirintului în care acesta va avea o ieşire, încercîndu-se ieşirea din labirint în timpul regulamentar. Versiunea propusă se poate complica, introducîndu-se în labirint şi nişte «obiecte» pe care personajul bun va trebui să le culeagă în număr cît mai mare. În acest caz «răii» vor fi nişte paznici, iar scopul jocului va fi culegerea de cît mai multe obiecte. De asemenea, pe măsura scurgerii timpului, labirintul îşi poate restrînge posibilităţile de parcurs, îngreunînd sarcina jucătorului. În sfîrşit, altă variantă ar fi existenţa a două tipuri de «obiecte» de cules, pentru unele acordîndu-se mai multe puncte. În această stuaţie, jucătorul va trebui să încerce să-şi optimizeze parcursul prin labirint.

Descrierea programului

10 (GOSUB 3000) — apelarea subrutinei de definire a caracterelor grafice pentru «bun» şi pentru «răi».

3000 — datele pentru caracterul grafic «bun».

3010 — datele pentru caracterul grafic «rău».

3020—3070 — ciclu de citire şi memorare a datelor care definesc caracterele «bun» şi «rău».

15 (GOSUB 1000) — apelarea subrutinei de trasare a labirintului.

1000— pereţii vor fi verzi.

1005—1130 — ciclu pentru calculul locului şi lungimii pereţilor. Acestea vor fi întîmplătoare.

1020, 1030 — calculul extremităţii peretelui (întîmplătoare, dar între 1 şi 30 pe orizontală şi 1 şi 20 pe verticală).

1040 — perete vertical: calculul celeilalte extremităţi a peretelui, cu limită la 20, lungimea cuprinsă între 1 şi 3.

1050 — acelaşi lucru pentru peretele orizontal.

1070—1110 — ciclu pentru trasarea peretelui.

1140 — trasare perete margine superioară. Se foloseşte tasta **8** împreună cu **CAPS SHIFT** în modul grafic.

1150 — trasare perete margine laterală.

1170 — trasare perete margine inferioară.
20 — rezervarea de memorie pentru fişierul poziţiilor celor 3 «răi» (variabila **c** — pentru coloană, variabila **l** — pentru linie).
25 — introducerea numărului de răi, **ng**.
26 — refuzul răspunsurilor defectuoase.
30 (GOSUB 900) — apelarea subrutinei pentru afişarea «răilor» şi «bunului». La începutul jocului pornesc din aceleaşi poziţii fixe.
900 — poziţia orizontală (**x**) şi verticală (**y**) a bunului.
905 — numărul de paşi (**pas**) la început este **0** (nu s-a făcut nici o mişcare).
910 — afişare «bun» — tasta **A** în modul grafic. Afişarea se face cu negru pe alb.
920 — ciclu pentru afişarea «răilor» (1, 2 sau 3, după alegerea făcută în linia 25).
940 — poziţia orizontală şi verticală a unui «rău».
950 — afişare «rău» — tasta **B** în modul grafic. Afişarea se va face cu albastru, roşu sau purpuriu (primul — albastru, al doilea — roşu, al treilea — purpuriu).
32 — semnalul de începere.
40—85 — ciclu pentru cronometrarea unei partide (**s** — unitate de măsură a timpului afişat).
50 — posibilitate de deplasare a unui «bun».
60—75 — ciclu pentru fiecare «rău».
65 — deplasare a unui «rău».
70 — posibilitate de deplasare a unui «bun».
80 — afişarea cronometrajului.
95, 100 — terminarea timpului regulamentar.
110 — ciclu pentru neutralizarea acţionărilor de taste care intervin imediat după gong.
120, 130 — sfîrşit.
500—670 — subrutină de deplasare a «bunului» în funcţie de tasta acţionată de jucător.
500 — memorarea tastei acţionate.
510 — nu s-a acţionat nici o tastă: întoarcerea la punctul de apel (linia 50 sau 70).
530 — mărimea iniţială a deplasării pe orizontală (**dx**) şi pe verticală (**dy**).
540 — saltul la o linie de program în funcţie de tasta acţionată.
545 — pregătirea deplasării spre stînga.
546 — pregătirea deplasării în sus.
547 — pregătirea deplasării în jos.
548 — pregătirea deplasării spre dreapta.
580, 590 — calculul viitoarei poziţii orizontale şi verticale.
610, 620 — respingerea posibilităţii de ieşire din labirint (ecran).
630 — respingerea posibilităţii de traversare a unui perete.
634 — ştergerea «bunului».
635 — a fost efectuat un pas; numărul paşilor (**pas**) creşte cu o unitate.
640, 650 — poziţia viitoare devine poziţia actuală.
660 — afişarea «bunului» în poziţia actuală. Pentru «bun» se foloseşte tasta **A** în modul grafic.
700—870 — subrutină de deplasare a unui «rău».
700, 710 — determinarea deplasării orizontale şi verticale în direcţia «bunului», lăsînd o parte de hazard: 0,1 sau 2 căsuţe (celule caracter) cu posibilitate de salt.
730—740 — viitoarea poziţie a «răului» (**xgn** — coordonata pe orizontală, **ygn** — coordonata pe verticală).
750, 760 — respingerea posibilităţii de ieşire din labirint (ecran).
770 — respingerea posibilităţii de aterizare pe un perete.
775 — ştergerea «răului» din poziţia sa actuală.
780, 790 — memorarea noii poziţii.
800 — afişarea «răului» cu albastru, roşu sau purpuriu, în funcţie de numărul său, la noua poziţie. Pentru caracterul grafic «rău» se foloseşte tasta **B** în modul grafic.
810, 820 — dacă «bunul» nu se găseşte în acelaşi loc cu «răul», jocul continuă.
830 — «bunul» se găsea în acest loc; afişarea mesajului şi a numărului de paşi.
840 — ciclu pentru neutralizarea acţionărilor de taste care intervin după oprirea jocului.
860, 870 — sfîrşit.

```
  5 BORDER 6: PAPER 7: INK 4
  7 CLS
  8 PRINT "LABIRINT"
 10 GO SUB 3000
 15 GO SUB 1000
 20 DIM c(3): DIM l(3)
 25 BEEP 0.2,20: INPUT "1,2 sau
 3 rai ";ng
 26 IF ng<1 OR ng>3 THEN GO TO
25
 27 LET ng=INT ng
 30 GO SUB 900
 32 BEEP 3,5: BEEP 0.2,10
 40 FOR s=0 TO 96 STEP ng+1
 50 GO SUB 500
 60 FOR g=1 TO ng
 65 GO SUB 700
 70 GO SUB 500
 75 NEXT g
 80 PRINT AT 21,29;s
 85 NEXT s
 95 BEEP 1,15
100 PRINT AT 21,0;"Salvat de go
ng! ";pas;" pasi"
110 FOR d=1 TO 200: NEXT d
120 INK 0
130 STOP
500 LET f$=INKEY$
510 IF f$="" THEN RETURN
530 LET dx=0: LET dy=0
535 IF f$<"5" OR f$>"8" THEN RE
TURN
540 GO TO 540+CODE f$-48
545 LET dx=-1: GO TO 580
546 LET dy=1: GO TO 580
547 LET dy=-1: GO TO 580
548 LET dx=1: GO TO 580
580 LET xn=x+dx
590 LET yn=y+dy
610 IF xn<0 OR xn>31 THEN RETUR
N
620 IF yn<0 OR yn>21 THEN RETUR
N
630 IF POINT (xn*8,(21-yn)*8)=1
 THEN RETURN
634 PRINT AT y,x;" "
635 LET pas=pas+1
640 LET x=xn
650 LET y=yn
660 INK 0: PRINT AT y,x;"<a>"
670 RETURN
700 LET dx=SGN (x-c(g))*INT (RN
D*3)
710 LET dy=SGN (y-l(g))*INT (RN
D*3)
730 LET xgn=c(g)+dx
740 LET ygn=l(g)+dy
750 IF xgn<0 OR xgn>31 THEN RET
URN
760 IF ygn<0 OR ygn>21 THEN RET
URN
770 IF POINT (xgn*8+4,(21-ygn)*
8+4)=1 THEN RETURN
775 PRINT AT l(g),c(g);" "
780 LET c(g)=xgn
790 LET l(g)=ygn
800 INK g: PRINT AT l(g),c(g);"
<b>"
810 IF x<>c(g) THEN RETURN
820 IF y<>l(g) THEN RETURN
830 BEEP 0.2,10: BEEP 0.3,5: PR
INT AT 21,0;"Ai fost mincat din
";pas;" pasi!";
840 FOR d=1 TO 200: NEXT d
860 INK 0
870 STOP
900 LET x=5: LET y=11
905 LET pas=0
910 INK 0: PRINT AT y,x;"<a>"
920 FOR g=1 TO ng
940 LET c(g)=20: LET l(g)=INT (
16*g/ng)
950 INK g: PRINT AT l(g),c(g);"
<b>"
960 NEXT g
970 RETURN
1000 INK 4
1005 FOR t=1 TO 30
1010 LET k=INT (RND*2)
1020 LET x1=INT (RND*30)+1
1030 LET y1=INT (RND*20)+1
1040 IF k=0 THEN LET x2=x1: LET
y2=y1+INT (RND*3)+1: IF y2>20 TH
EN LET y2=20
1050 IF k=1 THEN LET y2=y1: LET
x2=x1+INT (RND*3)+1: IF x2>30 TH
EN LET x2=30
1070 FOR x=x1 TO x2
1080 FOR y=y1 TO y2
1090 PRINT AT y,x;"<CAPS 8>"
1100 NEXT y
1110 NEXT x
1130 NEXT t
1140 PRINT AT 0,0;"<CAPS 8888888
8888888888888888888888888>"
1150 FOR l=1 TO 20: PRINT AT l,0
;"<CAPS 8>": PRINT AT l,31;"<CAP
S 8>": NEXT l
1170 PRINT AT 21,0;"<CAPS 888888
88888888888888888888888888>"
1180 RETURN
3000 DATA 60,66,129,165,129,255,
36,36
3010 DATA 60,126,255,255,255,255
,36,36
3020 FOR j=0 TO 1
3030 FOR i=0 TO 7
3040 READ a
3050 POKE USR CHR$ (65+j)+i,a
3060 NEXT i
3070 NEXT j
3080>RETURN
```

ROBAC

După apăsarea unei taste, pe ecran apare o rîmă de dimensiuni mai mari — ROBAC. Acesta va trebui condus pe suprafața de joc marcată cu o culoare mai deschisă, cu scopul de a mînca cît mai mulți păianjeni (gîndaci). Sînt 4 feluri de păianjeni care apar în mod întîmplător pe ecran în timpul jocului, iar pentru fiecare păianjen mîncat de ROBAC, jucătorul primește 10 puncte. Deplasarea lui ROBAC se va realiza prin intermediul tastelor: **A** — deplasare în jos; **Q** — deplasare în sus; **O** — la stînga; **P** — la dreapta. În timpul jocului (tot întîmplător) apar pe ecran și ciuperci otrăvitoare, care îl omoară pe ROBAC, dacă acesta le mănîncă. În acest caz, ROBAC, va pierde o viață (are în total 5 vieți) și jocul se va relua de la început, însă cu punctajul acumulat pînă în acel moment și cu viețile care i-au mai rămas la dispoziție. După pierderea tuturor celor 5 vieți, jocul se va putea relua (eventual de alt jucător) prin acționarea oricărei taste, de data aceasta, însă punctajul va porni de la 0. Atingerea unei margini a suprafeței de joc va marca o pierdere de 10 puncte, iar în acest caz, dacă nu se va acționa o tastă pentru identificarea direcției de deplasare, se vor pierde puncte în continuare și jocul se va pierde definitiv (cînd se ajunge la un punctaj negativ). El se poate relua de la început prin acționarea oricărei taste.

În partea de jos a ecranului apare afișat numărul de vieți ale lui ROBAC: cu cît acesta va fi de dimensiuni mai reduse (mai scurt) cu atît numărul de vieți care mai rămîn la dispoziție este mai mic. Tot în partea de jos a ecranului (dreapta) apare afișat și numărul de puncte realizat.

Modificări posibile. Jocul se poate complica prin următoarele adăugări sau modificări:

— introducerea nivelelor de dificultate. Un nivel de dificultate sporit va însemna apariția mai rapidă a ciupercilor față de păianjeni și/sau deplasarea mai rapidă a lui ROBAC;

— acordarea de puncte, diferențiat, în funcție de tipul păianjenului. În acest caz jucătorul va trebui să-și optimizeze traseul încercînd să-l determine pe ROBAC să mănînce mai repede păianjenii mai « prețioși »;

— la atingerea marginilor suprafeței de joc, se vor pierde mai multe puncte (toate sau o parte din ele);

— creșterea dificultății jocului pe măsură ce se pierd din vieți: apariția mai rapidă a ciupercilor față de păianjeni.

Pe aceeași idee se poate modifica și scenariul jocului ROBAC: în loc de o rîmă poate fi un personaj, de exemplu Făt Frumos (caracter grafic pentru om), care culege mere fermecate sau omoară balauri, ce apar din loc în loc pe ecran. Pe Făt Frumos îl poate urmări un zmeu care, dacă îl prinde, îl omoară. În acest caz, spre deosebire de ROBAC, în care ciupercile apar pe ecran, dar stau pe loc, va trebui realizată și deplasarea zmeului care îl urmărește pe Făt Frumos.

Descrierea programului

10 (GOSUB 6000) — apelarea subrutinei pentru descrierea lui ROBAC.

6010 — citirea datelor pentru descrierea lui ROBAC (linia 9700).

6020—6050 — definirea caracterelor grafice pentru corpul lui ROBAC (guri, cozi) și a păianjenilor.

6060 — variabila **r$** este o variabilă șir de caractere care conține toate caracterele grafice pentru gura deschisă. Ele se obțin astfel: gură deschisă în sus — tasta **A** în modul grafic; gură deschisă spre stînga — tasta **B** în modul grafic; gură deschisă spre dreapta — tasta **C** în modul grafic; gură deschisă în jos — tasta **D** în modul grafic.

6070 — variabila **g$** este o variabilă șir de caractere, care conține toate caracterele grafice pentru gură închisă. Ele se obțin astfel: gură închisă în sus — **E** în modul grafic; gură închisă spre stînga — **F** în modul grafic; gură închisă spre dreapta — **G** în modul grafic; gură închisă în jos — **H** în modul grafic.

6080 — variabila **o$** este o variabilă șir de caractere care conține toate caracterele grafice pentru coadă. Ele se obțin astfel: coada, cînd deplasarea se face spre stînga — **J** în modul grafic; coada, cînd deplasarea se face spre dreapta — **K** în modul grafic; coada cînd deplasareaa se face în jos — **L** în modul grafic.

6090 — **b$** este o variabilă șir de caractere, care conține toate caracterele grafice pentru păianjeni. Cele 4 feluri de păianjeni se obțin cu tastele: **N**, **P**, **O și Q** în modul grafic.

20 — alocarea spațiului de memorie pentru descrierea lui ROBAC. El poate avea o dimensiune maximă de 10 caractere grafice (2 cozi, 7 bucăți de corp și o gură sau, o coadă, 8 bucăți de corp și o gură).

90 — **a$** este o variabilă șir de caractere cu ajutorul căreia se va afișa ROBAC în partea de jos a ecranului. Aici el este de dimensiuni mai mici decît ROBAC-ul care se mișcă pe suprafața de joc. Caracterele grafice pentru obținerea ROBAC-ului mai mic se vor realiza cu tastele **S** pentru coadă, **T** pentru o bucată de corp și **U** pentru cap, toate în modul grafic. ROBAC-ul din partea de jos a ecranului va reprezenta numărul de vieți pe care le are ROBAC la un anumit moment dat. Inițial el este format din 3 bucăți; după pierderea unei vieți, o bucată va dispare, apoi încă una, apoi o jumătate din ultima bucată etc.

100 — inițializarea cu **0** a variabilei pentru punctaj (puncte).

110 — începe pregătirea suprafeței de joc.

130 (GOSUB 5000) — apelarea subrutinei pentru afișarea punctajului și a vieților în partea de jos a ecranului.

140 — inițializarea variabilei **paian.** Aceasta este un indicator care arată dacă ROBAC a mîncat un păianjen (**0**=nu, **1**=da).

150, 160 — inițializarea variabilelor pentru mărimea lui ROBAC. Corpul (gura și corpul propriu-zis) este format din 9 părți (caractere grafice) — variabila **cap**, plus o parte pentru coadă (variabila **coadă**).

170 — ROBAC începe să circule. Mișcarea lui se realizază astfel: pornește dintr-un punct fix spre dreapta avînd 2 cozi, 7 bucăți de corp și gura deschisă; se șterge o coadă; se închide gura; se înlocuiește gura cu o bucată de corp; se pune o gură deschisă pe celula caracter din față; se șterge coada și ciclul se repetă.

210, 220 — stabilirea coordonatelor.

230 — stabilirea direcțiilor în care este condus ROBAC. Dacă variabila **x1** este diferită de **0**, mișcarea se va face pe orizontală (s-a acționat tasta **0** — pentru deplasare stînga — sau **P** — pentru deplasare dreapta), iar dacă variabila **y1** este diferită de **0**, mișcarea se va face pe verticală (s-a acționat tasta **Q** — pentru deplasarea în sus — sau **A** — pentru deplasarea în jos).

270 — stabilirea coordonatelor lui ROBAC pentru deplasarea pe orizontală.

280 — stabilirea coordonatelor lui ROBAC pentru deplasarea pe verticală.

290 — testarea atingerii marginilor suprafeței de joc.

295 — se indică sonor atingerea marginii și se diminuează puncatjul cu 10 puncte. Dacă acesta a devenit negativ, jocul ia sfîrșit (1080).

298 — dacă nu s-a atins marginea, se continuă înaintarea.

300—320 — avansarea capului și a cozii.

340, 350 — stabilirea direcției de deplasare.

370 — ce s-a aflat în fața lui ROBAC?
380 — în fața lui ROBAC a fost o ciupercă; se pierde o viață.
390 (GOSUB 3000) — apelarea subrutinei pentru deplasarea lui ROBAC.
3010, 3020 — se arată mai întîi capul.
3030—3070 — corpul. Caracterul pentru corp din linia 3060 se obține cu tasta **M** în modul grafic. 3080, 3090 — coada.
400 (GOSUB 4000) — în fața lui ROBAC a fost un păianjen.
4010 — punctajul crește cu 10 puncte.
4020 — note muzicale.
4030 — se apelează subrutina pentru afișarea din partea de jos a ecranului (punctaj și vieți).
4040 — variabila **paian** redevine **0**.
410 (GOSUB 2000) — apelarea subrutinei pentru afișarea pe ecran a ciupercilor sau păianjenilor.
2020 — apare o ciupercă.
2030 — caracterul grafic pentru ciupercă se obține cu tasta **R** în modul grafic.
2040 — apare un păianjen.
2080 — nu apare nimic.

430 — de la început pentru a se verifica ce tastă s-a acționat pentru deplasarea lui ROBAC.
1010 — după ce a mîncat ciuperca, ROBAC va deveni pentru o secundă clipitor și va pierde o viață.
1040 — diminuarea lungimii lui ROBAC afișat în partea de jos a ecranului.
1050 — apelarea subrutinei pentru afișarea în partea de jos a ecranului.
1070 — se testează dacă mai sînt vieți. Dacă da, jocul se reia de la început cu punctajul actual (linia 110), dacă nu, se începe secvența de final de joc.
1100 — mesajul de sfîrșit de joc.
1120—1180 — melodia de sfîrșit de joc.
9500—9900 — date.
9500 — date ce reprezintă începutul drumului pentru ROBAC.
9600, 9610 — date pentru notele muzicale.
9700—9790 — date pentru definirea caracterelor pentru gurile lui ROBAC.
9800—9890 — date pentru definirea caracterelor pentru cozi.
9900 — date pentru definirea caracterului cozii.

```
   7 PRINT AT 10,8;"Asteapta un
pic..."
  10 GO SUB 6000
  20 DIM p(3,10)
  30 INK 6: BORDER 4: PAPER 6
  40 CLS
  50 PRINT #1;"Apasa o tasta"
  60 PAUSE 0
  80 CLS
  90 LET a$="<sttu sttu sttu>"
 100 LET puncte=0
 110 CLS
 120 PRINT #1;AT 1,0; PAPER 7;"
                              "
;#2
 130 GO SUB 5000
 140 LET paian=0
 150 LET cap=9
 160 LET coada=1
 170 RESTORE 9500
 180 FOR i=1 TO 9
 190 READ p(2,i),p(1,i),p(3,i)
 200 NEXT i
 210 LET x=1
 220 LET y=0
 230 LET x1=(INKEY$="p")-(INKEY$
="o")
 240 LET y1=(INKEY$="a")-(INKEY$
="q")
 250 IF x1<>0 THEN LET x=x1: LET
 y=0
 260 IF y1<>0 THEN LET y=y1: LET
 x=0
 270 LET x2=p(1,cap)+x
 280 LET y2=p(2,cap)+y
 290 IF x2>31 OR x2<0 OR y2>21 O
R y2<0 THEN GO TO 295
 293 GO TO 300
 295 BEEP .1,-10: LET puncte=pun
cte-10: IF puncte<0 THEN GO TO 1
079
 298 GO TO 390
 300 LET cap=cap+1
 310 IF cap>10 THEN LET cap=1
 320 LET coada=coada+1
 330 IF coada>10 THEN LET coada=
1
 340 LET p(1,cap)=x2
 350 LET p(2,cap)=y2
 360 LET p(3,cap)=x+2*y+3
 370 IF ATTR (y2,x2)=48 THEN LET
 paian=1
 380 IF ATTR (y2,x2)=50 THEN GO
TO 1000
 390 GO SUB 3000
 400 IF paian THEN GO SUB 4000
 410 GO SUB 2000
 420 PRINT AT p(2,cap),p(1,cap);
```

```
1;" PUNCTE ";puncte;#2
5020 RETURN
6010 RESTORE 9700
6020 FOR i=USR "a" TO USR "u"+7
6030 READ a
6040 POKE i,a
6050 NEXT i
6060 LET r$="<ab cd>"
6070 LET q$="<ef gh>"
6080 LET o$="<ij kl>"
6090 LET b$="<npoq>"
6100 RETURN
9500 DATA 5,5,4,5,6,4,5,7,4,5,8,
4,5,9,4,5,10,4,5,11,4,5,12,4,5,1
3,4
9600 DATA .4,12,.3,14,.1,12,.4,9
,.4,9,.3,9,.1,7,.3,9,.1,10,.6,9
9610 DATA .4,0,.3,2,.1,0,.4,-3,.
4,-3,.3,-3,.1,-5,.3,-3,.1,-2,.6,
-3
9700 DATA 66,195,195,231,255,255
,126,60
9710 DATA 60,126,31,15,15,31,126
,60
9720 DATA 60,126,248,240,240,248
,126,60
9730 DATA 60,126,255,255,231,195
,195,66
9740 DATA 44,110,239,239,239,255
,126,60
9750 DATA 60,126,255,255,7,255,1
26,60
9760 DATA 60,126,255,255,224,255
,126,60
9770 DATA 60,126,255,239,239,239
,110,44
9780 DATA 60,126,126,126,60,60,2
4,24
9790 DATA 0,112,124,255,255,124,
112,0
9800 DATA 0,14,63,255,255,63,14,
0
9810 DATA 24,24,60,60,126,126,12
6,60
9820 DATA 60,126,255,255,255,255
,126,60
9830 DATA 129,90,60,255,24,60,66
,129
9840 DATA 0,153,221,60,255,60,90
,129
9850 DATA 36,153,126,24,255,24,1
26,129
9860 DATA 36,36,153,126,36,36,10
2,153
9870 DATA 56,124,254,254,16,16,1
6,16
9880 DATA 0,28,62,126,255,126,62
,28
9890 DATA 0,115,255,255,255,255,
255,115
9900 DATA 0,144,254,252,248,252,
254,144
```

```
 INK 1;q$(p(3,cap));
 430 GO TO 230
1010 FLASH 1
1020 GO SUB 3000
1030 FLASH 0
1040 LET a$=a$(5 TO )+"    "
1050 GO SUB 5000
1060 BEEP 2,-10
1070 IF a$<>"                 " THE
N GO TO 110
1080 CLS
1090 INK 2
1100 PRINT AT 5,11;"AI MURIT!";A
T 10,10-LEN (STR$ puncte)/2;"PUN
CTE : ";puncte
1110 RESTORE 9600
1120 FOR I=1 TO 2
1130 FOR J=0 TO 9
1140 READ timp,ton
1150 BEEP timp,ton
1160 NEXT j
1170 PAUSE 20
1180 NEXT i
1190 PRINT AT 20,9;"tasteaza ENT
ER"
1200 PAUSE 0
1210 GO TO 30
2010 RANDOMIZE
2020 LET ik=2
2030 LET s$="<r>"
2040 IF RND<.3 THEN LET s$=b$(IN
T (4*RND)+1): LET ik=0
2050 LET tx=INT (21*RND)
2060 LET ty=INT (32*RND)
2070 IF ATTR (tx,ty)=48 THEN RET
URN
2080 IF ATTR (tx,ty)<>54 THEN LE
T ik=7: LET s$=" "
2090 PRINT AT tx,ty; INK ik;s$;
2100 RETURN
3010 PRINT AT p(2,cap),p(1,cap);
 INK 1;r$(p(3,cap));
3020 LET a=cap
3030 FOR i=1 TO 7
3040 LET a=a-1
3050 IF a<1 THEN LET a=10
3060 PRINT AT p(2,a),p(1,a); INK
 1;"<m>";
3070 NEXT i
3080 PRINT AT p(2,coada),p(1,coa
da); INK 1;o$(p(3,coada));
3090 LET a=coada-1
3100 IF a<1 THEN LET a=10
3110 PRINT AT p(2,a),p(1,a);" ";
3120 RETURN
4010 LET puncte=puncte+10
4020 BEEP .2,5: BEEP .1,10: BEEP
 .05,2
4030 GO SUB 5000
4040 LET paian=0
4050 RETURN
5010 PRINT #0;AT 1,0; INK 0; PAP
ER 7;" ROBAC ";a$;"      ";AT 1,2
```

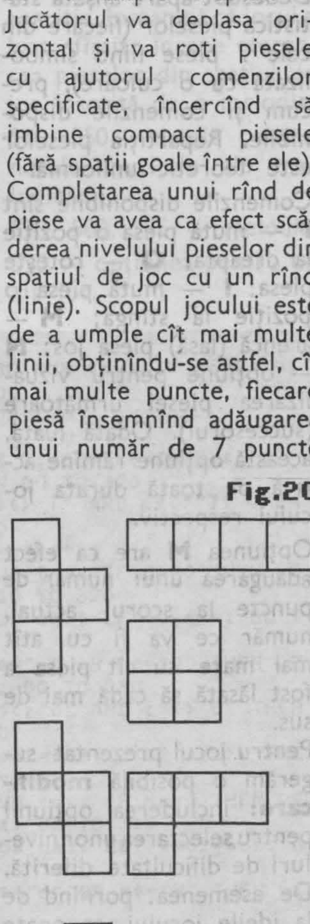
TETRIS – JOCUL COMBINĂRII PIESELOR

Jocul a apărut mai întîi în variante clasice fără calculator și a fost descris în multe lucrări, inclusiv în cartea «Între matematică și jocuri» [10], fiind și comercializat de RECOOP sub denumirea de Pentamino. Jocul constă în abilitatea de a îmbina mai multe piese geometrice alcătuite din cuburi de dimensiuni mai mici și de a forma astfel, diverse construcții. Dacă piesele sînt formate din 4 unități de bază (cubulețe), atunci jocul se numește Tetramino, iar dacă piesele sînt formate din 5 unități de bază, Pentamino, putînd de asemenea exista și Hexamino (cu 6 unități de bază). Cu cît numărul de unități de bază din care sînt formate piesele este mai mare, cu atît numărul pieselor crește și implicit și posibilitățile de combinare a lor devin mai complexe.

Transpunerea pe calculator a jocului a modificat într-un fel datele problemei și a pornit de la ideea semnalată și în lucrarea mai sus-amintită: «este însă nepractic să studiem prea mult aceste construcții, în sensul că, în loc să facem pentru fiecare teorie, este mai ușor să rezolvăm problema prin încercări, intuitiv, exploatînd restricțiile adhoc ale construcției, unele vizibile cu ușurință». Astfel, Tetris (varianta pe calculator a lui Tetramino) este practic un joc de îndemînare și reflexe, dar care dezvoltă abilitatea de a îmbina cît mai armonios piesele într-un interval de timp scurt. Nu mai avem de-a face cu piese formate din cubulețe (cu toate că și o versiune în tridimensional a Tetrisului nu ar fi imposibilă), ci cu piese geometrice plane pentru care unitățile de bază sînt pătratele. Este vorba de 7 piese (pătrate, dreptunghiuri, L-uri) formate din compunerea a cîte 4 pătrate unite între ele, astfel încît toată piesa să poată fi parcursă prin mișcări ale unui turn de șah (vezi fig. 20).

În partea dreaptă a ecranului apare delimitat un spațiu de joc în formă de cilindru, piesele (de diverse forme și culori) căzînd una cîte una în partea de jos a acestui spațiu, care se va umple treptat. Jucătorul va deplasa orizontal și va roti piesele cu ajutorul comenzilor specificate, încercînd să îmbine compact piesele (fără spații goale între ele). Completarea unui rînd de piese va avea ca efect scăderea nivelului pieselor din spațiul de joc cu un rînd (linie). Scopul jocului este de a umple cît mai multe linii, obținîndu-se astfel, cît mai multe puncte, fiecare piesă însemnînd adăugarea unui număr de 7 puncte

Fig.20



la scor. Cînd spaţiul de joc se umple cu piese, jocul se termină şi se cîntă o melodie, în timp ce spaţiul de joc este golit (şters). Jocul se poate relua la dorinţa jucătorului.

În partea stîngă-sus a ecranului apare afişat permanent scorul maxim posibil (MAX = 3000), scorul actual (AC) şi numărul de linii complete realizate (LINII).

Dedesubt apare afişată statistica pieselor (fiecare din cele 7 piese fiind simbolizată cu o culoare), precum şi comenzile disponibile. Repartiţia pieselor este teoretic uniformă.

Comenzile disponibile sînt **P** — mută piesa o poziţie la dreapta; **O** — roteşte piesa; **I** — mută piesa o poziţie la stînga; **M** — aruncă (lasă) piesa jos; **N** — opţiune pentru vizualizarea piesei următoare (succesorul). Odată luată, această opţiune rămîne activă pe toată durata jocului respectiv.

Opţiunea **M** are ca efect adăugarea unui număr de puncte la scorul actual, număr ce va fi cu atît mai mare cu cît piesa a fost lăsată să cadă mai de sus.

Pentru jocul prezentat sugerăm o posibilă **modificare:** includerea opţiunii pentru selectarea unor niveluri de dificultate diferită. De asemenea, pornind de la ideile jocului, se poate încerca realizarea unui joc **Pentis** (cu 12 piese formate din 5 unităţi de bază) şi a unuia **Hex** (cu piese formate din 6 unităţi de bază). În sfîrşit, se poate realiza şi un joc **Mix** în care să se combine toate piesele din cele 3 jocuri **Tetris, Pentis** şi **Hex.**

Descrierea programului

14—15 — rezervarea spaţiului de memorie pentru variabilele programului: **r** este un vector pentru generarea piesei. S-a rezervat spaţiu de memorie pentru 14 piese, deoarece fiecare piesă poate apărea de 2 ori (vezi-linia 1460); **s** contorizează statistica pieselor (de cîte ori apare fiecare din cele 7 piese); **u** şi **v** conţin descrierea piesei curente. Fiecare piesă este formată din 4 pătrăţele.

u (4,1) reprezintă coordonata pe orizontală a pătrăţelelor care descriu piesa.. Această coordonată se adună la coordonata piesei de referinţă.

u (4,4) reprezintă faptul că piesa poate fi pusă în 4 poziţii diferite.

v — acelaşi lucru pentru coordonata pe verticală. Cînd se ia decizia pentru o piesă, se citesc datele din DATA, se formează vectorii **x** şi **y** de descriere a pieselor, iar pentru piesa curentă se umplu **u** şi **v.** Vectorii **x** şi **y** au cîte 3 dimensiuni: prima reprezintă numărul piesei (max. 7 piese), a 2-a; reprezintă poziţia (coordonata pe orizontală şi respectiv pe verticală) a piesei, iar a 3-a dimensiune, reprezintă descrierea locului pătrăţelelor din care este formată piesa. În vectorul **a** se ţine minte care suprafaţe din spaţiul de joc sînt ocupate. În spaţiul de joc pot fi maximum 23×15 poziţii ocupate.

20 (GOSUB 1000) — apelarea subrutinei pentru încărcarea vectorilor **x** şi **y.**

30 (GOSUB 2000) — apelarea subrutinei pentru afişarea informaţiilor de pe partea stîngă a ecranului; 2070—2110 — afişarea comenzilor disponibile. Apelarea subrutinei pentru scor (GOSUB 1800).

Iniţial se pleacă de la scorul **0**. Dacă scorul actual a depăşit scorul cel mai mare (**hi**) atunci scorul actual devine scorul maxim (**hi**).

40—60 — desenarea spaţiului de joc.

110 — bucla principală; **pv** reprezintă piesa viitoare. Va fi aleasă în mod aleator una din cele 14 piese.

s(p) — statistica piesei curente; creşte cu o unitate la fiecare apariţie a piesei respective.

130 — variabilele **xx** şi **yy** reprezintă coordonatele pe orizontală şi pe verticală ale piesei de referinţă, iar **vx** şi **vy** reprezintă vechile coordonate (vechiul **x** şi vechiul **y**).

140 — se copiază din **x** şi **y** în **u** şi respectiv **v.**

170 — variabila **poz** arată poziţia piesei (cît este de rotită). Astfel, **poz** poate lua 4 valori, de la 1 la 4. Se calculează poziţia viitoare a piesei, **vpoz.** Se apelează subrutina pentru citirea tastaturii şi ştergerea piesei de pe poziţia anterioară (GOSUB 1900).

Se apelează subrutina care încearcă să mute piesa conform comenzii citite de la tastură (GOSUB **try**).

180 — (GOSUB **tip**) — apelarea subrutinei pentru afişarea piesei în poziţia **xx, yy**.

200 — se citeşte tastatura; testarea se realizează de 4 ori, corespunzător celor 4 posibilităţi (K = 4) de mişcare a piesei (stînga, dreapta, rotire, cădere).

210 — se testează în primul rînd dacă s-a acţionat tasta **I**.

220 — dacă s-a acţionat tasta **I**, se începe mişcarea în stînga a piesei, scăzîndu-se coordonata pe orizontală cu o unitate (vechea coordonată pe orizontală — **vx** — rămîne cu o unitate mai mare decît coordonata **xx**).

230 — se apelează subrutina de testare a posibilităţii de mutare în stînga a piesei. Dacă testul este pozitiv, piesa se va şterge din poziţia respectivă.

240 — apelarea subrutinei de ştergere a piesei (**del**) după care ciclul se reia.

250 — începe ciclul pentru testarea comenzii de deplasare la dreapta (tasta **P**). De această dată, coordonata pe orizontală creşte cu o unitate (vechea coordonată pe orizontală — **vx** rămîne cu o unitate mai mică decît coordonata **xx**).

260—290 — se aplică algoritmul asemănător pentru ştergerea piesei şi mutarea ei la dreapta, după ce s-a testat această posibilitate.

300 — verificarea dacă s-a acţionat tasta **N** pentru indicarea succesorului.

Variabila **nxt** este un indicator care arată dacă s-a acţionat sau nu tasta **N** (**nxt** = 1 înseamnă că s-a acţionat tasta **N**).

305 — verificare dacă s-a acţionat tasta **M** pentru opţiunea de lăsare jos a piesei.

350 — calculul scorului şi afişarea lui.

5000 — cînd piesa nu se mai poate mişca încă de la apariţie, înseamnă că jocul a luat sfîrşit. Începe ştergerea pieselor din spaţiul de joc însoţită de o melodie.

510, 550 — datele pentru notele muzicale ale melodiei.

520, 580, 585 — melodia.

590 — opţiunea pentru un alt joc.

```
  10 RANDOMIZE : PAPER 0: INK 7:
 BORDER 0: BRIGHT 1: CLS : LET t
itlu=1000: LET imagine=2000
  15 DIM r(14): DIM s(7): DIM u(
4,4): DIM v(4,4): DIM a(23,15):
DIM x(7,4,4): DIM y(7,4,4)
  20 GO SUB titlu
  30 CLS : GO SUB imagine: GO SU
B 1800
  40 FOR i=1 TO 21
  50 LET a(i+1,14)=8: LET a(i+1,
2)=8: PRINT AT i,17; FLASH 1;CHR
$ 134;AT i,29;CHR$ 134;: BEEP 0.
01,8
  60 NEXT i
  70 FOR i=18 TO 28
  80 LET a(2,i-15)=8: PRINT AT 2
1,i; FLASH 1;"<6>";: BEEP 0.01,8
  90 NEXT i
 110 LET pv=r(INT (1+RND*14))
 120 LET p=pv: LET pv=r(INT (1+R
ND*14)): LET s(p)=s(p)+1: PRINT
AT p+8,7;s(p);
 130 LET xx=6: LET yy=19: LET vx
=6: LET vy=19
 140 FOR i=1 TO 4: FOR j=1 TO 4
 150 LET u(i,j)=x(p,i,j): LET v(
i,j)=y(p,i,j)
 160 NEXT j: NEXT i
 170 LET poz=1: LET vpoz=1: GO S
UB 1900: GO SUB try: IF da=0 THE
N GO TO 500
 180 GO SUB tip
 190 FOR k=1 TO 4
 200 PAUSE 2: LET c$=INKEY$: IF
c$="" THEN GO TO 320
 210 IF c$<>"i" AND c$<>"I" THEN
 GO TO 250
 220 LET xx=xx+1
 230 GO SUB try: IF da=0 THEN LE
T xx=vx: GO TO 320
 240 GO SUB del: LET vx=xx: GO T
O 310
 250 IF c$="p" OR c$="P" THEN LE
T xx=xx-1: GO TO 230
 260 IF c$<>"o" AND c$<>"O" THEN
 GO TO 300
 270 LET poz=poz+1: IF poz>4 THE
N LET poz=1
 280 GO SUB try: IF da=0 THEN LE
T poz=vpoz: GO TO 310
 290 GO SUB del: LET vpoz=poz: G
O TO 310
 300 IF c$="n" OR c$="N" THEN LE
T nxt=1
 305 IF c$="m" OR c$="M" THEN GO
 TO 470
 310 LET k=k+1: IF k>4 THEN LET
k=4
 320 NEXT k
```

```
 330 LET yy=yy-1: GO SUB try: IF
 da=0 THEN GO TO 350
 340 GO SUB del: LET vy=yy: GO T
O 190
 350 LET scor=scor+5+(1-nxt)*2:
BEEP 0.01,8: PRINT AT 3,6;scor;:
 LET yy=vy: FOR i=1 TO 4: LET a(
yy+2+v(poz,i),xx+2+u(poz,i))=p:
NEXT i: LET w=0
 360 LET l=yy: LET q=0
 370 FOR j=3 TO 13
 380 IF a(l,j)=0 OR a(l,j)=8 THE
N LET j=14: NEXT j: GO TO 450
 390 NEXT j: LET q=q+1: LET line
=line+1: PRINT AT 4,6;line;: LET
 scor=scor+50: PRINT AT 3,6;scor
 400 LET k=0: FOR j=3 TO 13
 410 LET a(l,j)=a(l+1,j): IF a(l
,j)=0 THEN LET k=k+1
 420 PRINT AT 23-l,31-j; PAPER a
(l,j);" ";: BEEP 0.005,8
 430 NEXT j
 440 IF k<11 THEN LET l=l+1: GO
TO 400
 450 LET w=w+1: IF q=0 THEN LET
yy=yy+1
 455 IF w<4 THEN GO TO 360
 460 GO TO 120
 470 LET yy=yy-1: GO SUB try: IF
 da=0 THEN LET yy=yy+1: GO SUB d
el: LET vy=yy: GO TO 350
 480 LET scor=scor+1: PRINT AT 3
,6;scor;: BEEP 0.008,1: GO TO 47
0
 500 FOR i=17 TO 29: PRINT AT 0,
i; FLASH 1;"<6>";: BEEP 0.01,8:
NEXT i: FOR i=16 TO 21: PRINT AT
 i,8;"              ": NEXT i: PRINT
AT 21,8;"Jcul s-a sfirsit";: LET
 sc=168: INK 0:: FOR j=1 TO 2: R
ESTORE 510
 510 DATA 0,12,16,12,2,11,11,19,
0,12,16,12,2,11,11,11,0,12,16,12
,5,17,16,14,7,12,-5,11,0,12
 520 FOR i=1 TO 30: READ nota: B
EEP 0.25,nota: LET sc=sc-1: PLOT
 145,sc: DRAW 86,0: NEXT i: BEEP
 0.5,0: NEXT j
 530 FOR j=1 TO 2: RESTORE 540
 540 DATA 7,17,17,17,7,16,14,12,
5,11,11,11,4,7,12,16,7,17,17,17,
7,16,14,12,-5,7,7,7,0,12
 550 FOR i=1 TO 30: READ nota: B
EEP 0.25,nota: LET sc=sc-1: PLOT
 145,sc: DRAW 86,0: NEXT i: BEEP
 0.5,0: NEXT j
 560 FOR j=1 TO 2: RESTORE 570
 570 DATA -3,9,12,16,-3,9,12,16,
-8,14,-8,11,-3,12,11,9,-3,9,12,1
6,-3,9,12,16,-8,14,-8,11,-3,9
 580 FOR i=1 TO 30: READ nota: B
EEP 0.25,nota: LET sc=sc-1: IF s
c>7 THEN PLOT 145,sc: DRAW 86,0
```

```
 585 NEXT i: BEEP 0.5,0: NEXT j:
 INK 7
 590 CLS : PRINT AT 12,6:"Inca o
 data?(d/n)";: LET t=0
 591 LET c$=INKEY$: LET t=t+1: I
F c$="" AND t<1000 THEN GO TO 59
1
 592 IF c$="" OR (c$<>"n" AND c$
<>"N") THEN DIM a(23,15): CLS :
GO SUB imagine: GO SUB 1800: GO
TO 40
 593 RANDOMIZE USR 0
 600 LET da=1: FOR e=1 TO 4
 610 IF a(yy+2+v(poz,e),xx+2+u(p
oz,e))<>0 THEN LET da=0
 620 NEXT e: RETURN
 800 FOR e=1 TO 4
 810 PRINT AT 21-vy-v(vpoz,e),29
-vx-u(vpoz,e);" ";
 813 NEXT e: FOR e=1 TO 4
 815 PRINT AT 21-yy-v(poz,e),29-
xx-u(poz,e); PAPER p;" ";
 820 NEXT e: RETURN
 950 GO TO 800
1000 CLS
1010 RESTORE 1020
1020 DATA 0,0,0,0,-1,0,1,2
1030 DATA -1,0,1,2,0,0,0,0
1040 DATA 0,0,0,0,1,0,-1,-2
1050 DATA 1,0,-1,-2,0,0,0,0
1070 DATA 1,0,-1,-1,0,0,0,-1
1080 DATA 0,0,0,-1,-1,0,1,1
1090 DATA -1,0,1,1,0,0,0,1
1100 DATA 0,0,0,1,1,0,-1,-1
1120 DATA 1,0,-1,-1,0,0,0,1
1130 DATA 0,0,0,1,-1,0,1,1
1140 DATA -1,0,1,1,0,0,0,-1
1150 DATA 0,0,0,-1,1,0,-1,-1
1170 DATA 1,0,0,-1,0,0,-1,0
1180 DATA 0,0,-1,0,-1,0,0,1
1190 DATA -1,0,0,1,0,0,1,0
1200 DATA 0,0,1,0,1,0,0,-1
1220 DATA 1,0,0,-1,0,0,-1,-1
1230 DATA 0,0,-1,-1,-1,0,0,1
1240 DATA -1,0,0,1,0,0,1,1
1250 DATA 0,0,1,1,1,0,0,-1
1270 DATA 1,0,0,-1,0,0,1,1
1280 DATA 0,0,1,1,-1,0,0,1
1290 DATA -1,0,0,1,0,0,-1,-1
1300 DATA 0,0,-1,-1,1,0,0,-1
1320 DATA 1,0,1,0,0,0,1,1
1330 DATA 0,0,1,1,-1,0,-1,0
1340 DATA -1,0,-1,0,0,0,-1,-1
1350 DATA 0,0,-1,-1,1,0,1,0
1380 FOR i=1 TO 7
1390 FOR j=1 TO 4
1400 FOR k=1 TO 4: READ x(i,j,k)
: NEXT k
1410 FOR k=1 TO 4: READ y(i,j,k)
: NEXT k
1420 NEXT j
1430 NEXT i
1460 DATA 1,2,3,4,5,6,7,1,2,3,4,
5,6,7
```

```
1470 FOR i=1 TO 14: READ r(i): N
EXT i
1480 LET scor=0: LET line=0: LET
 hi=3000: LET try=600: LET del=8
00: LET tip=950
1490 LET t=0
1495 LET c$=INKEY$: LET t=t+1: I
F t<800 AND c$="" THEN GO TO 149
5
1496 RETURN
1800 IF scor>hi THEN LET hi=scor
1805 PRINT AT 2,6;hi
1810 LET scor=0: LET line=0: DIM
 s(7)
1820 PRINT AT 3,6;scor;AT 4,6;li
ne;: FOR i=1 TO 7: PRINT AT i+8,
7;s(i);
1830 NEXT i
1840 LET nxt=0: RETURN
1900 IF nxt<>1 THEN RETURN
1910 FOR e=1 TO 4
1920 PRINT AT 18-y(p,1,e),11-x(p
,1,e);" ";
1930 NEXT e
```

```
1940 FOR e=1 TO 4
1950 PRINT AT 18-y(pv,1,e),11-x(
pv,1,e); PAPER pv;" ";
1960 NEXT e
1970 RETURN
2000 INK 6: PRINT AT 0,0;"T E T
R I S"
2010 PRINT AT 2,0;"MAX";AT 3,0;"
AC";AT 4,0;"LINII";
2020 PRINT AT 7,0;"Statistica"
2030 FOR i=1 TO 7
2040 PRINT AT 8+i,1; PAPER i;" "
; PAPER 0;"...."
2050 NEXT i
2060 PRINT AT 21,11;"SUCC";
2070 PRINT AT 17,0;"I-stinga"
2080 PRINT AT 18,0;"P-dreapta"
2090 PRINT AT 19,0;"O-rotatie"
2100 PRINT AT 20,0;"M-cade"
2110 PRINT AT 21,0;"N-succesor"
2120 RETURN
9990 CLEAR : SAVE "tetris" LINE
10
9995 VERIFY "tetris"
```

Jocuri de aventură

În timp ce toate jocurile logice (solitare sau competitive) şi multe dintre jocurile de reflexe pot fi practicate şi cu alte mijloace decît cu un calculator (a se vedea « jocurile mecanice » din sălile cu această destinaţie), jocurile didactice şi jocurile de aventură *sînt tipice pentru calculator. Am numit « de aventură » acele jocuri în care, intrînd în rolul unui personaj anumit, trebuie să duceţi la bun sfîrşit o sarcină relativ complexă care presupune gestionarea adecvată a unor resurse, în sensul larg al cuvîntului, alegerea permanentă între mai multe variante de acţiune, propuse de program, prevenirea unor pericole şi aşa mai departe. Se cer astfel implicate o bună analiză a situaţiilor în care vă aflaţi, decizii echilibrate, intuiţie, uneori şi puţin noroc. Multe dintre jocurile de acest gen sînt veritabile jocuri de* întreprindere*, similare celor folosite cu mare succes în pregătirea/testarea persoanelor care se pregătesc pentru funcţii de decizie economică. Deosebirea este că de data aceasta obiectivul aparent al jocului are o tentă ludică pronunţată, fiind vorba despre drumuri în junglă, comerţ interstelar, administrarea resurselor unui imperiu istoric, confruntat cu diferite pericole interne sau externe, căutarea unei comori într-o peşteră şi aşa mai departe. Multe dintre jocurile de aventură renunţă la imagine, desfăşurîndu-se la nivelul unui text-dialog, constînd în descrierea situaţiei curente, întrebări, răspunsuri şi evaluări ale deciziilor. Adesea intervin şi restricţii de timp, cerînd de data aceasta rapiditate în gîndire, în alegerea acţiunii de întreprins, nu în executarea ei, ca la jocurile de reflexe. Mai puţin spectaculoase ca înfăţişare, jocurile de aventură devin însă pasionante după intrarea în « hainele » personajului (ceva asemănător cu identificarea cu un personaj dintr-o aventură literară, dintr-un roman, de pildă). Încercaţi-le pe cele care urmează şi vă veţi convinge.*

PIERDUT ÎN JUNGLĂ

Jucătorul se află în mijlocul junglei, avînd la dispoziție cîteva resurse pentru supraviețuire: o busolă (pentru a se putea orienta), un pistol cu 6 gloanțe, un cuțit și o praștie (pentru a se putea apăra). Prin deciziile pe care le ia, va trebui să străbată jungla și să iasă cu bine din ea, ajungînd la o așezare omenească. Jucătorul își poate alege un joc ușor, potrivit sau greu. În funcție de nivelul de dificultate ales, probabilitatea de apariție a unei situații speciale (leu, păianjen uriaș, mlaștină, nisipuri mișcătoare, crocodil etc.) crește. Jungla are o formă pătrată (100×100 km), punctul din care pornește inițial jucătorul fiind ales întîmplător. Jocul se desfășoară sub formă de dialog prin care calculatorul descrie situația în care se află jucătorul și opțiunile pe care acesta le are la dispoziție și cere luarea unei decizii de către jucător. Deciziile jucătorului pot fi **de deplasare** (taste folosite: **A** — pentru nord, **B** — pentru sud, **C** — pentru est și **D** — pentru vest), jucătorul parcurgînd la fiecare opțiune o distanță de 1 km, sau **de ieșire** dintr-o situație specială descrisă în prealabil de calculator (leu, păianjen, etc.). Schema de desfășurare a jocului se poate urmări în fig. 21. Se observă cum deciziile inițiale sînt cele de deplasare, putînd apoi apare (în mod aleator, în funcție de gradul de dificultate ales), situații speciale (pe nivelul 1) care, la rîndul lor, pot genera alte situații speciale (pe nivelul 2). După fiecare decizie a jucătorului, calculatorul evaluează locul în care se găsește jucătorul în cazul în care a fost o decizie de deplasare, sau starea (puterea) jucătorului în cazul în care a fost o decizie de rezolvare a unei situații speciale. Jocul poate lua sfîrșit după evaluarea locului în care se află jucătorul (dacă acesta a ieșit din junglă) sau după evaluarea stării sale (dacă « puterea » sa a ajuns la 0). Jocul se poate relua, eventual de către alt jucător.

Descrierea programului

1 — modificări ale variabilelor de sistem pentru realizarea unui dialog fără

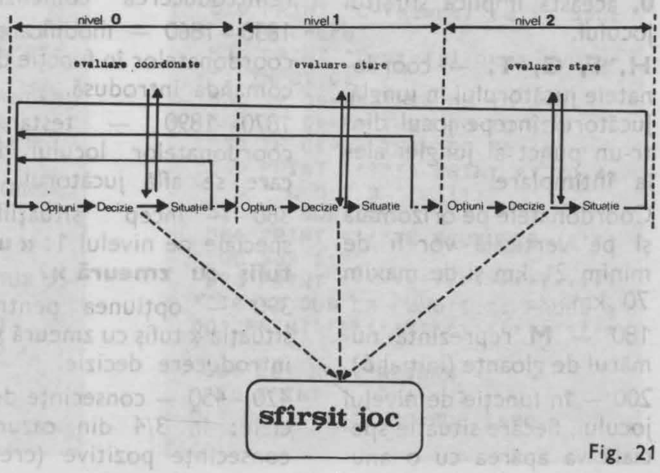


Fig. 21

mesajul « scroll? » și numai cu litere mari (cursor C).
30—80 — prezentare sumară a jocului.
110—120 — introducerea gradului de dificultate. Variabila **J$** reprezintă gradul de dificultate introdus de către jucător **(U, M** sau **G)**. Această valoare (șir de caractere) se transformă într-o valoare numerică (prin intermediul variabilei **J** cu care se va lucra în program și care va reprezenta gradul de dificultate.
160 — comenzile disponibile, reprezentate prin variabilele: **A$** (pentru nord), **B$** (pentru sud), **C$** (pentru est) și **D$** (pentru vest). Variabilele **F$** și **G$** reprezintă alte șiruri de caractere folosite pentru dialog (în situații speciale numărul comenzilor disponibile poate fi mai mare).
170 — **K** reprezintă puterea jucătorului; aceasta crește cînd jucătorul bea apă, mănîncă zmeură etc. și scade cînd așteaptă, se rănește etc. Inițial **K** este **100** iar cînd ajunge la **0**, aceasta implică sfîrșitul jocului.
H, F, G, T, — coordonatele jucătorului în junglă. Jucătorul începe jocul dintr-un punct al junglei ales la întîmplare.
Coordonatele pe orizontală și pe verticală vor fi de minim 21 km și de maxim 70 km.
180 — **M** reprezintă numărul de gloanțe (inițial **6**).
200 — în funcție de nivelul jocului, fiecare situație specială va apărea cu o anumită probabilitate. Variabila **Y** va reprezenta valoarea cu ajutorul căreia se va selecta numărul de linie la care se generează o situație.
220—340 — stabilirea punctului de pornire a jocului (linia de program) și generarea diferitelor situații în funcție de gradul de dificultate introdus.
350 — situație obișnuită (jucătorul a parcurs 1 km în direcția indicată); puterea scade cu o unitate. Dacă **K** ajunge la 0, jocul ia sfîrșit (linia 2050), nemaiapelîndu-se rutina (GO SUB 1760). prin care se generează dialogul pentru o situație obișnuită.
1760 — mesajul și cererea de indicare a comenzii.
1770 — tipărirea comenzilor disponibile.
1780—1810 — selectarea unei linii (în funcție de comanda introdusă) pentru modificarea coordonatelor locului din junglă în care se află jucătorul.
1820 — dacă se acționează o tastă care nu reprezintă o comandă validă se cere reintroducerea comenzii.
1830—1860 — modificarea coordonatelor în funcție de comanda introdusă.
1870—1890 — testarea coordonatelor locului în care se află jucătorul.
380 — încep situațiile speciale pe nivelul 1: **« un tufiș cu zmeură »**.
390 — opțiunea pentru situația « tufiș cu zmeură »; introducere decizie.
420—450 — consecințe decizie: în 3/4 din cazuri, consecințe pozitive (crește puterea jucătorului), în 1/4 din cazuri, consecințe negative (zmeură otrăvitoare).
460 — calcului situației jucătorului **(K)**.
470 — situație specială: **« nisipuri mișcătoare »**.
480—520 — descriere opțiuni pentru situația specială « nisipuri mișcătoare »; introducere decizie (520).
530—550 — selecatare consecință în funcție de opțiune.
560—730 — consecințe.
750 — situație specială: **« leu »** (sau **« șarpe »**).
760—820 — descriere opțiuni pentru situația specială; introducere decizie.
830—870 — selectare consecințe în funcție de opțiune.
880—1200 — consecințe.
1220 — situație specială: **« paianjen uriaș pe ceafă »**.
1230—1260 — descriere opțiuni pentru situația specială; introducere decizie **(Q$)**.
1270—1300 — selectarea unei ramuri de program în funcție de decizia luată.
1310—1380 — consecințe posibile ale deciziei luate.
1350 — dacă s-a luat decizia « strigi », în 1/3 din cazuri paianjenul va mușca, iar în 2/3 din cazuri va pleca.
1430 — situație specială: **« mlaștină »**.
1440—1460 — descriere opțiuni pentru situația specială; introducere decizie **(Q$)**.
1470—1490 — selectarea unei ramuri de program în funcție de decizia luată.

1510—1540 — consecințe posibile ale deciziei luate în care aceasta a fost « bei apă ».
1510 — în 1/3 din cazuri (RND< 0.3) va rezulta o consecință negativă, iar în 2/3 din cazuri una pozitivă.
1610 — situație specială de nivel 2: **« crocodil »** (în anumite cazuri, dacă în prealabil s-a luat decizia de trecut mlaștina înot).
1630—1650 — descriere opțiuni (consecințe) ale apariției crocodilului (întîmplătoare).
1690 — situație specială: **ciocnire de un copac, leu sau șarpe.** Este în funcție de **Y** (1710, 1720, 1730).
1910—1930 — jucătorul a îndeplinit scopul jocului și a ieșit cu viață din junglă.
1940 — opțiunea de joc nou.
1950 — jucătorul s-a apropiat de o așezare omenească.
1960—1970 — jucătorul se aproprie de marginea junglei.

```
   1 POKE 23692,255: POKE 23658,
8
   3 FOR G=1 TO 25: PRINT : NEXT
 G
  20 RANDOMIZE
  30 BORDER 5: PAPER 6: INK 1: C
LS : PRINT '"      Pierdut in
 jungla"
  40 PRINT FLASH 1;"<CAPS 333333
33333333333333333333333333>": GO
 SUB 2110
  50 PRINT '"      Te-ai ratacit
 in jungla"
  60 PRINT "Ai un pistol cu sase
 gloante,un   cutit,o prastie si
 o busola ."
  70 PRINT '''"  TREBUIE SA IESI
 CU VIATA DIN             JUNGL
A"
  80 PRINT '"      Jungla are 100x
100 Km."
 100 PRINT '': POKE 23692,255
 110 PRINT "Vrei un joc:"'"U-uso
r;"'"M-mediu;"'"G-greu?"
 120 PRINT : INPUT J$
 130 LET J$=J$(1): LET J=CODE J$
: LET J=(J-50)*J/80
 140 PRINT '''" APASA O TASTA CA
 SA PORNESTI!": PAUSE 0
 160 LET A$="A": LET B$="B": LET
 C$="C": LET D$="D": LET E$="E":
 LET F$="F": LET G$="G"
 170 LET K=100: LET H=INT (RND*5
0+21): LET F=INT (RND*50+21)
 180 LET M=6: LET G=100-F: LET I
=100-H: LET Z$=C$
 200 LET S=J-10: LET Y=INT (RND*
S)
 210 GO SUB 1190
 220 IF Y=1 THEN GO SUB 470
 230 IF Y=2 OR Y=3 THEN GO SUB 7
50
 240 IF Y=4 THEN GO SUB 1220
 250 IF Y=5 THEN GO SUB 1430
 260 IF Y=6 THEN GO SUB 380
 270 IF Y=7 THEN GO TO 350
 280 IF Y=8 THEN GO TO 350
```

```
 290 IF Y=9 THEN GO TO 350
 300 IF Y=10 THEN GO TO 350
 310 IF Y=11 THEN GO TO 350
 320 IF Y=12 THEN GO TO 350
 330 IF Y=13 THEN GO TO 350
 340 IF Y=14 THEN GO SUB 1690
 350 LET K=K-1: IF K>0 THEN GO S
UB 1750
 355 IF K<=0 THEN GO TO 2050
 360 GO TO 190
 380 PRINT '"Ai gasit un tufis c
u zmeura"
 390 PRINT "Le maninci?"'"A-da;"
'"B-nu": INPUT Q$
 400 IF Q$=A$ THEN GO TO 450
 410 RETURN
 420 IF RND<.75 THEN GO TO 450
 430 PRINT "ERAU OTRAVITOARE !!!
"'"Te-ai imbolnavit"
 440 LET I=INT (1-RND): RETURN
 450 PRINT "Ti-ai mai potolit fo
amea"
 460 LET K=INT (K+J/10): RETURN
 470 PRINT '"Nisipuri miscatoare
!!!": GO SUB 1190
 480 PRINT "Ce faci?"
 490 PRINT "A-te zbati sa iesi;"
 500 PRINT "B-stai nemiscat;"
 510 PRINT "C-incerci sa te apuc
i de o craca"
 520 PRINT "D-strigi dupa ajutor
;": INPUT Q$
 530 IF Q$=B$ THEN GO TO 690
 540 IF Q$=C$ THEN GO TO 650
 550 IF Q$=D$ THEN GO TO 610
 560 PRINT '''''': PRINT AT 20,4;
"  B ^ L D ^ B ^ C !!!"
 570 GO SUB 1190
 580 PRINT '''"TE SCUFUNZI !!!"'
: GO SUB 1190
 590 PRINT ''''''''''''''''''''''
'"TE-AI DUS LA FUND!!!": PAUSE 1
00: PRINT ''''''''''''''''''''''
'
 600 LET K=0: RETURN
 610 PRINT "    A    J    U    T
O    R    !!! ": GO SUB 1190
```

```
620 PRINT "      A  J  U  T  O
R !!": GO SUB 1190
630 PRINT "          A J U T O
R !": GO SUB 1190: IF Y=4 THEN R
ETURN
640 PRINT TAB 12;"AJU.....": GO
SUB 1190: PRINT TAB 14;"A...":
GO SUB 1190: PRINT TAB 14;"Ah.."
: GO SUB 1190: GO TO 590
650 IF RND*J>6 THEN GO TO 670
660 GO SUB 1190: PRINT "NU AI R
EUSIT !!!": GO SUB 1190: GO TO 5
80
670 GO SUB 1190: PRINT "AI REUS
IT !!!"
680 LET K=INT (K*(1-RND*.67)):
RETURN
690 PRINT : GO SUB 710: PRINT "
Au trecut ";Q;" ore.": IF Q<24 A
ND K>0 THEN PRINT "Acum ";: GO T
O 480
700 GO TO 1120
720 LET Q=INT (RND*40+1): LET K
=INT (K-Q)
730 PAUSE Q*10: RETURN
750 IF Y=2 THEN PRINT TAB 10;"u
n LEU !!!"
760 IF Y=3 THEN PRINT TAB 10;"u
n SARPE !!!"
770 PRINT "Ce faci?"
780 PRINT "A- o iei la sanatoas
a;"
790 PRINT "B- tragi cu pistolul
;"
800 PRINT "C- tragi cu prastia;
"
810 PRINT "D- folosesti cutitul
;"
820 PRINT "E- te urci in copac?
": INPUT Q$
830 IF Q$=A$ THEN GO TO 890
840 IF Q$=B$ THEN GO TO 970
850 IF Q$=C$ THEN GO TO 1020
860 IF Q$=D$ THEN GO TO 1030
870 IF Q$=E$ THEN GO TO 1080
880 LET K=INT (K-1): PRINT "Nu
s-a miscat din loc.": GO TO 770
890 GO SUB 1990: GO SUB 2040
900 IF Z$=Q$ THEN GO SUB 1690:
GO TO 770
910 IF RND*K<6 THEN GO TO 940
920 PRINT "Uff !!": GO SUB 1190
930 PRINT "Esti in siguranta.":
RETURN
940 PRINT "TE-A AJUNS DIN URMA
!!!"
950 LET K=INT (RND*K): IF K<5 T
HEN GO TO 1060
960 LET J=J-1: PRINT "Acum ": G
O TO 770
970 IF M<1 THEN PRINT "   CLIC
!...": PRINT "NU MAI AI GLOANTE
!!!": LET Q=17: GO TO 1050
980 GO SUB 1190: PRINT "
B A N G !!!": GO SUB 1190
990 LET L=J*2.5: LET M=M-1
1000 IF RND*L>17 THEN GO TO 1150
1010 PRINT "NU AI NIMERIT !!!":
GO TO 940
1020 LET L=J/1.5: GO TO 1040
1030 LET L=J/1.7: IF Y=J THEN LE
T L=L*2
1040 LET Q=INT (RND*L+1)
1050 IF Q>18 THEN GO TO 1150
1060 PRINT "          Te-a muscat
!": LET K=K-(2*Q)
1070 IF K<15 THEN PRINT "       T
E-A MINCAT !!!": LET K=0: RETURN

1075 GO TO 960
1080 IF Y=3 THEN GO TO 1130
1090 PRINT "Leul sta si asteapta
.": GO SUB 710: IF K<30 THEN GO
TO 1180
1100 PRINT "Au trecut ";Q;" ore.
": IF Q>24 THEN GO TO 1120
1110 PRINT "De acum poti cobori.
": RETURN
1120 GO SUB 1190: GO SUB 1190: G
O SUB 1190: PRINT "TI-E FOAME SI
 TI-E SETE !!!": LET K=0: RETURN

1130 PRINT "Serpii se urca in co
paci mai          usor ca oamenii
 !"
1140 PRINT "S-A INCOLACIT IN JUR
UL TAU !!!": GO SUB 710: PRINT "
      TE-A INGHITIT !!!": LET K=
0: RETURN
1150 PRINT "L-ai omorit !": GO S
UB 1190: GO SUB 1190
1160 PRINT "Il maninci :"'"A-da;
"'"B-nu ?": INPUT Q$: IF Q$=A$ T
HEN LET K=K+J
1170 RETURN
1180 PRINT '"AI CAZUT DIN COPAC
DE EPUIZARE !": GO TO 950
1190 LET W=INT (RND*88)+1: POKE
23692,255
1200 PAUSE W: RETURN
1220 PRINT "Un paianjen URIAS ti
 s-a asezat          pe ceafa !"
1230 PRINT "Ce faci :"'"A-ramii
nemiscat;"
1240 PRINT "B-te scuturi;"
1250 PRINT "C-il impusti;"
1260 PRINT "D-strigi ?": INPUT Q
$
1270 IF Q$=A$ THEN GO TO 1390
1280 IF Q$=B$ THEN GO TO 1350
1290 IF Q$=C$ THEN GO TO 1320
1300 IF Q$=D$ THEN GO TO 1340
1310 GO TO 1360
1320 IF M<1 THEN PRINT "      CL
IC !...": GO TO 1350
1330 PRINT '"       B A N G !!!"
: GO SUB 1190: PRINT "TE-AI IMPU
SCAT IN CEAFA !!!": GO TO 824
1340 GO SUB 610
1350 IF RND<.6 THEN GO TO 1380
```

```
1360 PRINT "          Te-a muscat
!": GO SUB 1190
1370 PRINT "        ERA VENINOS !!
!": GO SUB 1190: LET K=0: RETURN

1380 PRINT " Uff !  A plecat .":
 RETURN
1390 GO SUB 710: IF K<5 AND K>0
THEN PRINT "    Lesini de spaim
a !": GO TO 1350
1400 PRINT "Au trecut ";Q;" ore
.": IF Q>24 THEN GO TO 1120
1410 IF Q<5 THEN PRINT "Acum ";:
 GO TO 1230
1430 PRINT '"Ai ajuns la o mlast
ina."
1440 PRINT "Ce faci :"'"A-iti po
tolesti setea;"
1450 PRINT "B-treci inot mai aep
arte;"
1460 PRINT "C-te intorci ?": INP
UT Q$: GO SUB 1190
1470 IF Q$=A$ THEN GO TO 1510
1480 IF Q$=B$ THEN GO TO 1560
1490 IF Q$=C$ THEN GO TO 1670
1500 PRINT "Ai cazut in apa !":
GO TO 560
1510 IF RND<.3 THEN GO TO 1540
1520 LET K=K+RND*K/2
1530 PRINT "Te-ai mai racorit ."
: RETURN
1540 PRINT "ERA INFECTATA !!!"'"
Ai febra si esti foarte slabit."
1550 LET K=K-INT (RND*K): RETURN

1560 LET Q=INT (RND*4)+1: GO SUB
 1190: LET K=K-2
1570 IF Q=1 THEN GO TO 1600
1580 IF Q=2 THEN GO TO 1610
1590 GO SUB 1190: PRINT "Ai reus
it sa treci cu bine .": RETURN
1600 PRINT "    TE INNECI !!!": G
O TO 610
1610 PRINT " Un crocodil !": LET
 K=K-1: GO SUB 1190
1620 PRINT " INCA UNUL !!!": GO
SUB 1190: PRINT "Inoata mai repe
de !": GO SUB 1190
1630 LET Q=INT (RND*J)+1
1640 IF Q<20 THEN PRINT "   Te-a
 ajuns !!!": GO SUB 1190: PRINT
"        TE-A MINCAT !!!": LET
 K=0: RETURN
1650 PRINT "   Uff ! Ai ajuns la
 mal .": PRINT "Ai scapat doar c
u o sperietura !"
1660 LET K=K-INT (RND*K): RETURN

1670 GO SUB 1990: LET K=K-1: RET
URN
1690 PRINT '"            BUF !!!"'
1700 PRINT "Te-ai ciocnit de un
";
1710 IF Y=2 THEN PRINT "... LEU
!": RETURN
1720 IF Y=14 THEN PRINT "... COP
AC !": GO SUB 1190: LET K=INT (K
*.8): RETURN
1730 IF Y=3 THEN PRINT "... SARP
E !": RETURN
1740 RETURN
1760 PRINT TAB 10;"Ai mers 1 Km.
"'"INCOTRO O IEI ACUM :"
1770 PRINT "A-est;"'"B-vest;"'"C
-nord;"'"D-sud ?": INPUT X$: LET
 Z$=X$
1780 IF X$="A" THEN GO TO 1830
1790 IF X$="B" THEN GO TO 1840
1800 IF X$="C" THEN GO TO 1850
1810 IF X$="D" THEN GO TO 1860
1820 GO TO 1770
1830 LET F=F-1: LET G=G+1: GO TO
 1870
1840 LET F=F+1: LET G=G-1: GO TO
 1870
1850 LET H=H-1: LET I=I+1: GO TO
 1870
1860 LET H=H+1: LET I=I-1
1870 IF F=0 OR G=0 OR H=0 OR I=0
 THEN GO TO 1910
1880 IF F<5 OR G<5 OR H<5 OR I<5
 THEN GO TO 1960
1890 IF F<10 OR G<10 OR H<10 OR
I<10 THEN GO TO 1950
1900 RETURN
1910 PRINT "          AI REUSIT !
!!"
1920 PRINT "         ESTI SALVAT
!!!"
1930 PRINT "AI AJUNS LA O ASEZAR
E OMENEASCA."
1940 GO SUB 1190: PAUSE 0: PRINT
 ''''''''''''''''''''''''"MAI DORE
STE CINEVA SA INCERCE :"'"A-DA"'
"B-NU": INPUT O$: IF O$<>A$ THEN
 GO TO 2060
1945 GO TO 100
1950 PRINT " SE AUD OAMENI IN DE
PARTARE !!!": RETURN
1960 PRINT "       Esti aproape !"
1970 PRINT " SE ZARESC NISTE CAS
E !!!"
1980 RETURN
1990 IF Z$=A$ THEN LET X$=B$
2000 IF Z$=B$ THEN LET X$=A$
2010 IF Z$=C$ THEN LET X$=D$
2020 IF Z$=D$ THEN LET X$=C$
2030 RETURN
2040 PAUSE INT (RND*30/J): PRINT
 " si fugi !": LET K=INT (K*.9):
 GO SUB 1780: RETURN
2050 PRINT "               MORI ."'
"IMI PARE RAU,JOCUL S-A SFIRSIT.
"': GO TO 1940
2060 PRINT "APASA ORICE TASTA SA
 CONTINUI .": PAUSE 0: CLS
2070 POKE 23692,255: PRINT AT 20
,10;"LA REVEDERE !"''''''''''''
2100 BORDER 7: GO TO 9999
2110 RETURN
```

COMOARA DIN PEȘTERĂ

Un joc clasic de aventură în care jucătorul se plimbă într-un labirint (peșteră) încercînd să găsească o comoară ascunsă aici și să iasă cu ea afară din peșteră. Dar, pe lîngă faptul că aceasta este păzită cu strășnicie de uriaș, jucătorul are de înfruntat și o serie întreagă de alte primejdii: pirați, lilieci, un personaj ciudat (Bill Bones), care nu se arată niciodată, dar trece din loc în loc și pe unde lasă urme nu se mai poate înainta etc. Cu cît jucătorul înfruntă mai multe primejdii și reușește să ia comoara și să o scoată la lumină cu atît primește mai multe puncte. Pe de altă parte, cu cît va găsi un itinerar mai scurt cu atît va scoate comoara dintr-un număr mai mic de pași. Jocul se desfășoară printr-un dialog continuu între jucător (care ia decizii prin introducerea comenzilor) și calculator, care, după fiecare comandă, descrie situația și locul unde se află jucătorul.

Comenzile disponibile pentru jucător sînt:

N — merge la nord; **S** — merge la sud; **E** — merge la est; **V** — merge la vest; **U** — urcare; **C** — coborîre; **P** — punctaj.

Deci, are la dispoziție 6 comenzi pentru deplasare și o comandă prin care poate afla oricînd punctajul acumulat. Dacă se acționează o tastă care nu indică una din comenzile disponibile, atunci se va afișa din nou setul de comenzi disponibile, iar jucătorul se va menține în aceeași situație (loc).

În indicarea comenzilor, jucătorul are de înfruntat două tipuri de probleme. Primele sînt de natură logică. Dacă, de exemplu, după o comandă de mers spre nord **(N)** apare mesajul: «Tunelul cotește. În ce direcție mergi?», în acest caz, jucătorul va trebui să indice ca direcție est **(E)** sau vest **(V)**, dar în nici un caz nord **(N)** sau sud **(S)**. Fapt aparent banal, care poate da însă bătaie de cap jucătorilor neexperimentați. Alte probleme sînt legate de orientarea într-un labirint iar aceste probleme sînt mai dificile. De aceea, considerăm necesar explicarea unor amănunte legate de geografia peșterii, precum și unele indicații referitoare la modul de abordare a jocului de către copii mai mici sau jucători neexperimentați.

Peștera este organizată pe 3 niveluri (vezi figura 22): parter (simbolizat cu **0**), etaj

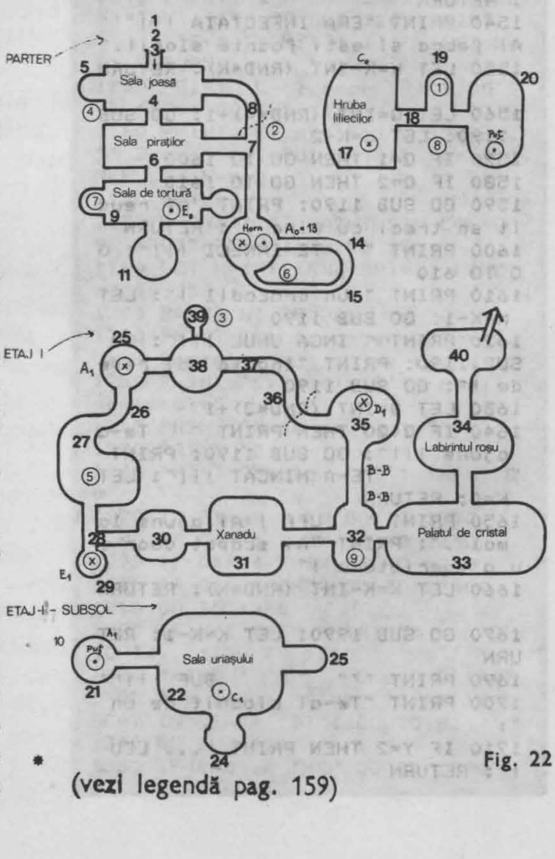


Fig. 22

(vezi legendă pag. 159)

(simbolizat cu **1**) și subsol (simbolizat cu —**1**). De la un nivel la altul se poate trece numai prin anumite puncte. Unui punct de urcare de la parter la etaj (de exemplu) îi va corespunde un punct de coborîre de la etaj la parter. Comoara nu va fi ascunsă în același loc la fiecare joc.

Pentru copiii mici, recomandăm utilizarea hărții peșterii (figura), iar pentru cei de 8—15 ani, desenarea planului peșterii pe măsură ce jocul avansează. În acest fel se vor evita situațiile în care se merge în cerc (se trece de mai multe ori prin aceleași puncte) și se va forma deprinderea de a realiza planuri și hărți după descrieri.

Modificări posibile. Deși în joc pare destul de complicată, peștera are numai 40 de locuri (încăperi), în plus existînd și suficientă memorie disponibilă pentru dezvoltarea jocului. Deci, se poate crește numărul încăperilor, a etajelor, a situațiilor etc. De asemenea, pe baza logicii programului se poate concepe un alt scenariu cu alte aventuri, în locul uriașului poate fi un balaur, în locul comorii poate fi o Ileană Cosînzeană furată etc. Ar fi interesantă și posibilitatea adăugării graficii: tablouri pentru fiecare (sau unele) încăpere a peșterii (lucru complicat, deoarece fiecare imagine-ecran ocupă o memorie de aproape 8 KO) sau desenarea (reconstituirea) automată a planului (hărții) peșterii, pe măsură ce ea este străbătută.

Descrierea programului

5 — dialog pe ecran fără mesajul « scroll ? ».

20 — rezervarea de spațiu de memorie pentru încăperile peșterii — variabila **A**, pentru încăperile (locurile) unde se pune comoara la începutul jocului **(P)** și a celor șapte comenzi disponibile **(C)**.

30—40 — asignarea variabilor pentru comenzile disponibile.

50 — inițializarea variabilelor: **KT** reprezintă numărul de comenzi (pași de program), pe care le-a dat jucătorul; **K** — reprezintă numărul încăperii (inițial se pornește de la 3, primii 3 pași fiind în pădure); **PT** reprezintă punctajul; **INC** — numărul încăperii în care se află jucătorul la un moment după indicarea unei comenzi; **MC** — numărul de încăperi.

60 — apelarea subrutinelor pentru cazul în care nu s-a indicat una din comenzile disponibile: GO SUB 300 — se vor afișa 4 rînduri goale fără mesajul « scroll? »; GO SUB 400 — se vor repeta comenzile disponibile.

66 — se citesc datele care indică locul unde este ascunsă comoara **P(I)**. Acestea se interpretează astfel: prima oară comoara se ascunde în încăperea nr. 19, a doua oară în încăperea nr. 7 etc.

80 — se citesc datele referitoare la încăperi **A(I,J)**. Fiecărei încăperi îi sînt asociate date care indică posibilitățile pentru fiecare comandă, precum și numărul de puncte pe care îl obține jucătorul trecînd prin acest loc.

85 — variabila **CO** este un indicator pentru comoară. Dacă **CO=0** comoara nu este în posesia jucătorului; **CO = 1** comoara este în posesia jucătorului. Cînd jucătorul găsește și ia comoara, primește un număr de puncte (la începutul jocului **50** de puncte).

100 — jocul începe cu o întrebare; se păstrează prima literă din răspuns (comandă) și se tipărește cu litere mari.

110 — numărul de comenzi date (pași) a crescut cu o unitate

965 — afișarea unui mesaj pentru fiecare comandă în funcție de încăperea în care se găsește jucătorul.

970—1095 — mesajele.

890—960 — datele care descriu peștera.

890 — datele referitoare la numărul încăperii în care se pune comoara: prima oară în camera 19, a doua oară în camera 7 etc.

900—960 — descrierea încăperilor peșterii. Fiecărei încăperi îi sînt asociate cîte 8 date (deci linia 900 conține descrierea primelor 6 camere), în următoarea ordine: Nord, Sud, Est, Vest, Urcare, Coborîre, numărul mesajului asociat fiecărei încăperi, punctajul pentru fiecare încăpere sau 10 în cazul în care comoara se află în încăperea respectivă. Pentru o deplasare, datele se interpretează după formula: **INC = INC+ data**.

Astfel, data **0** înseamnă că nu se poate merge în direcția asociată (**INC** — numărul încăperii rămîne același, deci jucătorul se va găsi în aceeași încăpere). De exemplu, cele 8 date asociate primei încăperi se citesc astfel:

0 — la Nord nu se poate merge;
2 — dacă se indică Sud, la numărul camerei actuale se adaugă 2 și se obține numărul camerei în care se ajunge;
0 — la Est nu se poate merge;
0 — la Vest nu se poate merge;
0 — nu se poate Urcare;
0 — nu se poate Coborîre;
3 — mesajul asociat camerei actuale este cel cu numărul **3**.

O dată negativă, de exemplu, prima dată **(—1)** pentru datele asociate celei de-a doua încăperi, se va interpreta astfel: numărul încăperii în care se ajunge dacă se va da comanda asociată (Nord) se va găsi adunînd —**1** la numărul camerei actuale **(2)**. Deci jucătorul va ajunge în camera **1**. A 24-a dată **(50)**, indică numărul de puncte obținut dacă se trece prin această cameră (**3** deoarece este ultima dată din al treilea set de 8 date).

```
  5 POKE 23692,255
 10 BORDER 5: PAPER 5: INK 2: F
LASH 0: OVER 0: INVERSE 0: BRIGH
T 0: CLS
 20 DIM A(40,8): DIM C$(7): DIM
 P(10)
 30 LET C$(1)="N": LET C$(2)="S
": LET C$(3)="E": LET C$(4)="V"
 40 LET C$(5)="U": LET C$(6)="C
": LET C$(7)="P"
 50 LET KT=0: LET K=3: LET PT=0
: LET INC=1: LET MT=26: LET MC=4
0
 60 GO SUB 300: GO SUB 400: GO
SUB 300
 66 FOR I=1 TO 10: READ P(I): N
EXT I
 70 GO SUB 300
 80 FOR I=1 TO MC: FOR J=1 TO 8
: READ A(I,J): NEXT J: NEXT I
 85 LET A(P(INC),8)=10: LET CO=
0
 90 LET T=A(K,7): GO SUB 500
100 PRINT ';TAB 5;"IN CE DIRECT
IE MERGI ?";: POKE 23692,255: GO
 SUB 102: LET A$=CHR$ ((CODE A$)
-32): PRINT A$
101 GO TO 110
102 PAUSE 0: LET A$=INKEY$
103 IF CODE A$<99 OR CODE A$>11
8 THEN LET A$="_"
104 RETURN
110 PRINT : LET KT=KT+1
120 FOR I=1 TO 7
130 IF A$=C$(I) THEN GO TO 160
140 NEXT I
150 GO SUB 400: GO TO 100
160 IF I<>7 THEN GO TO 170
162 IF PT<20 THEN LET P$=" PUNC
TE.": GO TO 166
165 LET P$=" DE PUNCTE."
166 PRINT "AI ";PT;P$: GO SUB 3
00: GO TO 100
170 LET N=A(K,I)
180 IF N=0 AND CO<>1 THEN PRINT
 TAB 7;"NU POTI ";A$
190 LET K=K+N: LET T=A(K,7): LE
T C=A(K,8)
200 GO SUB 600: GO SUB 500: GO
TO 100
300 POKE 23692,255: FOR J=1 TO
4: PRINT : NEXT J: RETURN
400 PRINT "  COMENZILE DISPONIB
ILE SINT :"
410 PRINT '';TAB 9;"N-NORD , S-
SUD"
420 PRINT ';TAB 9;"E-EST , V-VE
ST"
430 PRINT ';TAB 5;"U-URCARE , C
-COBORIRE"
440 PRINT ';TAB 11;"P-PUNCTAJ"
450 RETURN
500 GO TO 965
510 GO SUB 300
520 RETURN
600 IF C=0 THEN GO TO 640
610 IF INT (C/10)<>1 THEN GO TO
 650
620 GO SUB 300: PRINT " FELICIT
ARI! Comoara este aici!"'"   VRE
I SA O IEI ? (DA/NU)": PAUSE 0:
LET A$=INKEY$: PRINT
630 IF A$="D" OR A$="d" THEN LE
T CO=INC: LET PT=PT+50
640 LET PRU=0: RETURN
650 IF INT (C/10)<>2 THEN GO TO
 700
660 IF CO=0 THEN GO TO 690
670 PRINT '"    Este un tunel i
ngust."'"  Nu poti trece cu com
oara."
680 LET K=K-N: LET T=A(K,7): LE
T C=A(K,8)
690 LET PRU=0: RETURN
700 IF INT (C/10)<>3 THEN GO TO
 790
710 IF CO=0 THEN GO TO 780
720 LET PRU=PRU+1
730 LET T=T+1: IF C-INT (C/10)*
10=5 THEN LET T=MT-1
```

```
 740 IF PRU=1 THEN GO SUB 500
 750 IF PRU>1 THEN GO TO 770
 760 LET K=K-N: LET T=A(K,7): LE
T C=A(K,8): RETURN
 770 LET T=T+1: GO SUB 500: LET
A(P(INC),8)=40: LET INC=INC+1: L
ET PT=PT-20: LET CO=0: LET PRU=0
: IF INC>10 THEN LET INC=1
 775 LET A(P(INC),8)=10
 780 RETURN
 790 IF INT (C/10)=4 THEN PRINT
"Pe un bilet scrie :"'" Doar nu
crezi ca fac greseala    sa o as
cund in acelasi loc!";TAB 20;"BI
LL BONES": GO SUB 300
 800 IF INT (C/10)<>5 OR CO=0 TH
EN GO TO 860
 810 PRINT "              FELICITAR
I!"'"  AI REUSIT SA FURI COMOARA
 DIN "',KT;" PASI!"
 820 GO SUB 300: PRINT "  Mai vr
ea cineva sa incerce ?";TAB 13;"
(D/N)": PAUSE 0: LET A$=INKEY$
 830 IF A$<>"D" AND A$<>"d" THEN
 CLS : STOP
 840 FOR I=1 TO 10: LET A(P(I),8
)=0: NEXT I: LET PT=0: LET K=3:
LET INC=INC+1: LET CO=0: LET KT=
0: IF INC>10 THEN LET INC=1
 845 LET A(P(INC),8)=10
 850 RETURN
 860 IF INT (C/10)=6 THEN LET PT
=PT+20
 870 IF (INT (C/10)=7 AND (CO-IN
T (CO/2)*2=0)) THEN PRINT "Aceas
ta este o zona in curs de  creer
e interzisa vizitatorilor.": GO
TO 680
 880 RETURN
 890 DATA 19,7,39,5,27,16,10,36,
32,21
 900 DATA 0,2,0,0,0,0,3,1,-1,1,-
1,-1,0,0,2,0,-1,1,-1,-1,0,0,1,50
,-1,2,4,1,0,0,4,0,0,0,-1,0,0,0,5
,0,-2,3,1,0,0,0,7,35
 910 DATA 1,6,0,-1,0,0,8,0,0,-1,
0,-4,0,0,6,20,-3,2,3,1,20,0,9,0,
0,0,-1,0,0,0,5,0,-2,0,0,0,0,0,5,
0,0,0,0,-3,0,0,5,0
 920 DATA -6,3,1,0,12,8,10,60,1,
0,0,-1,0,0,6,0,0,-1,0,1,0,0,6,0,
-3,0,-1,0,0,0,6,35,0,0,1,0,0,5,2
1,0,1,0,2,-1,0,0,22,0
 930 DATA 0,-1,0,0,0,0,5,60,0,0,
0,-2,15,0,15,0,0,0,1,0,-8,0,15,0
,0,2,1,-1,-5,0,11,30,0,0,0,-1,0,
0,5,0,-2,0,0,0,0,0,5,0
 940 DATA 0,1,13,0,0,12,23,0,-1,
0,0,1,0,0,6,0,0,1,-1,0,0,0,20,0,
-1,1,2,0,0,0,22,0,-1,0,0,0,0,-20
,24,35,0,0,1,-2,0,0,20,0
 950 DATA 0,0,1,-1,0,0,19,60,3,0
,1,-1,0,0,14,60,1,0,0,-1,0,0,16,
70,0,-1,0,0,0,6,17,1,0,-3,0,1,0,
-15,14,0,1,0,-1,0,0,0,6,20
```

```
 960 DATA 0,-1,0,1,0,0,20,0,1,0,
-1,-13,0,0,22,0,0,-1,0,0,0,0,5,0
,-36,-36,-36,-36,-36,-36,18,50
 965 IF T<>0 THEN GO TO 965+T*5
 970 PRINT "Esti in padure ; int
rarea in    pestera este in sud.
": GO TO 520
 975 PRINT "Nu cred ca in felul
acesta vei  gasi pestera.": GO T
O 520
 980 PRINT "Te-ai ratacit in pad
ure.": GO TO 520
 985 PRINT "Esti intr-o sala joa
sa.Spre nordse vede lumina.": GO
 TO 520
 990 PRINT "Tunelul se infunda."
: GO TO 520
 995 PRINT "Esti intr-un tunel c
are coteste.": GO TO 520
1000 PRINT "Esti in sala piratil
or ; din    fericire sint plecat
i.": GO TO 520
1005 PRINT "Tunelul merge N-S ;
exista  o   ramificatie spre V":
 GO TO 520
1010 PRINT "Sala de tortura.Pe j
os sint     imprastiate schelete
.": GO TO 520
1015 PRINT "Treci pe linga un ho
rn.": GO TO 520
1020 PRINT "Esti in sala uriasul
ui.Calca    usor : doarme!": GO
TO 520
1025 PRINT "S-a trezit uriasul!
FUGI!!!": GO TO 520
1030 PRINT "Prichindelule,vrei s
a furi      comoara? Mai incearc
a!": GO TO 520
1035 PRINT "Pe zid scrie: BILL B
ONES A FOST AICI.": GO TO 520
1040 PRINT "Esti la baza unui pu
t.": GO TO 520
1045 PRINT "            Palatul de cr
istal .      Muzica izvoraste
 de pretutindeni": GO TO 520
1050 PRINT "  Ratacesti in Labir
intul Rosu.": GO TO 520
1055 PRINT "           Bine ai veni
t!": GO TO 520
1060 PRINT "In fata ta este FLUV
IUL MORTII. Pe o tablita scrie:
XANADU...": GO TO 520
1065 PRINT "Tunelul se largeste.
Sint gauri  in pereti.": GO TO 5
20
1070 PRINT "Esti in hruba liliec
ilor.       Grabeste-te!": GO T
O 520
1075 PRINT "Tunelul se ramifica.
": GO TO 520
1080 PRINT "S-a lasat o ceata de
asa . Din   cind in cind se aud
tipete.": GO TO 520
1085 PRINT "Esti in galeria came
rei de      tortura.Se vad urme
```

```
insingerate.": GO TO 520
1090 PRINT "Se aud piratii certi
ndu-se.FUGI!": GO TO 520
1095 PRINT "Aha! La tine este co
moara!    S-o punem la loc si
mai vorbim!    Nu fi prost!  FU
GI!": GO TO 520
```

ALTE JOCURI

Numeroase jocuri au rămas în afara claselor avute în vedere mai înainte, anume toate acelea la care hazardul are un rol hotărîtor. (Sigur, și la jocurile de reflexe, ba chiar și la cele de aventură și didactice se întîlnesc intervenții esențiale ale factorului aleator, dar nu în modul în care acționează jucătorul, ci în alegerea sarcinii curente pe care acesta o are de rezolvat; altfel spus, acțiunile jucătorului depind numai de el însuși și, desigur, de starea jocului în acel moment, hazardul intervenind numai în definirea acelei stări). Din punctul de vedere al antrenării inteligenței și a altor trăsături de acest tip, jocurile « de noroc » nu au o valoare deosebită. Trebuie însă operată o diferențiere între aceste jocuri, în funcție de ceea ce îi mai rămîne totuși jucătorului de făcut. Unele jocuri sînt « mai de noroc » decît altele. Să considerăm un exemplu: tablele. Un joc clasic. În 1979, campionul mondial la table, un italian, a fost învins de un calculator cu un scor categoric. 6 la 1. Un fapt semnificativ, care arată că tablele sînt un joc rațional, pentru că un calculator nu joacă la întîmplare! Sau, mai corect spus, asta dovedește că pentru a juca bine table trebuie efectuate calcule probabiliste exacte, pe care oamenii le fac probabil din instinct și experiență, putîndu-se astfel contracara influența (mare, evident) hazardului. Bineînțeles, nu pledăm aici nici pentru table, nici pentru alte jocuri similare, ci pentru afirmația că aceste jocuri nu trebuie ignorate sau negate pur și simplu, pentru că multe dintre ele prezintă componente raționale semnificative, iar marea majoritate sînt realmente atractive (și răspîndite). Atractive pentru dorința de aventură, de necunoscut din psihologia omului, care este atras nu numai de limpezimile raționamentului, dar și de fiorul indus de clarobscurul neprevăzutului, al « baftei ». Nu am ales însă aici decît cîteva jocuri de această factură, două (popularul « șeptică » și mult-răspînditul în rîndul copiilor, « spînzurătoare ») nefiind jocuri pure de hazard (primul cere evaluări de probabilități, iar programul face acest lucru cînd joacă, al doilea bazîndu-se pe o bună cunoaștere a limbii române — din partea jucătorului).

Jocul este destul de răspîndit în rîndul copiilor, care îl numesc « spînzurătoare ». Unul din jucători (în cazul nostru, calculatorul, alege un cuvînt, despre care nu dezvăluie decît cîte litere are, iar adversarul trebuie să-l ghicească, propunînd în mod repetat cîte o literă. Dacă litera se găsește în cuvîntul ascuns, o dată sau de mai multe ori, cel care a ales cuvîntul indică locurile respective. Dacă litera nu apare în cuvînt, se desenează o « piesă » a unei spînzurători. Jocul continuă în felul acesta, pînă ce fie cuvîntul a fost identificat, fie se desenează întreaga spînzurătoare, după care omulețul agățat de ea este « spînzurat ». Programul care urmează are un dicționar de 25 de cuvinte, de lungimi cuprinse între 2 și 9 litere, din care alege un cuvînt. Spînzurătoarea este compusă din 8 părți, deci, la a noua literă eronată partida este pierdută. După « spînzurare », cuvîntul ascuns este făcut cunoscut. Literele propuse sînt precizate permanent pe ecran. Dacă o literă care apare în cuvînt este propusă în mod repetat, ea va fi din nou scrisă pe ecran, fără a constitui eroare (dar după umplerea a două rînduri, va apărea eroare de execuție).
La încheierea unei partide, jocul poate fi reluat.
Desigur, o **modificare** de interes poate fi introducerea unui set mai bogat de cuvinte, pentru a mări gradul de dificultate a jocului. Atenție însă, lungimea cuvintelor nu trebuie să depășească 9 litere, altfel sînt necesare schimbări mai detaliate în program.

Descrierea programului

10 — **v** prezintă literele ghicite, iar **c$** conține fondul de cuvinte din care calculatorul alege.

20 — 30 — citirea matricei **c$**

40 — cadrul desenat.

60 — **x** este coordonata de bază a omulețului.

70 — gura omulețului.

80 — se alege la întîmplare un cuvînt.

100—120 — se calculează lungimea cuvîntului (variabila **b**).

130 — **q** este o variabilă-indicator, pentru a permite scrierea literelor propuse de jucător pe două linii.

150 — **c** este numărul de litere propuse, iar **d** este numărul de încercări infructuoase.

160—180 — se indică lungimea cuvîntului pe ecran.

190—200 — se propune o literă.

205 — **ind = 0** indică o încercare infructuoasă, **ind = 1** indică ghicirea unei litere.

230—240 — dacă s-au propus peste 30 de litere, cele care urmează se scriu pe rîndul următor (**q = 1**).

250—270 — se caută litera în cuvîntul ascuns.

280 — se scriu literele deja ghicite.

300 — dacă s-a ghicit o literă se continuă.

410 — a fost făcută a noua eroare, jocul se încheie.

420—450 — se mai adaugă o piesă la spînzurătoare.

500 — ghicire reușită.

510—530 — omulețul este șters din partea dreaptă.

540—570 — omulețul este desenat în stînga.

500—630 — cade trapa.
640—730 — omulețul este spînzurat.
740 — se dezvăluie cuvîntul.
800—830 — opțiune de reluare.
1000—1001 — se desenează omulețul, luînd ca bază de plecare valoarea lui **x** (în dreapta sau în stînga spînzurătorii).
2000—2030 — coordonatele și lungimile pieselor spînzurătorii (sînt folosite la liniile 420—450).

```
   5 BORDER 1: PAPER 6: CLS
  10 DIM v$(10): DIM c$(25,10)
  20 FOR i=1 TO 25: READ c$(i):
NEXT i
  30 DATA "cocostirc","vacanta",
"om","profesor","substanta","arg
ument","inghetata","hipopotam","
artilerie","invatator","oaie","g
irafa","chimie","oras","stea","a
lpinist","african","var","siluet
a","joc","doc","geometrie","alfa
bet","cuvint","fotbal"
  40 PLOT 93,52: DRAW 150,0: DRA
W 0,110: DRAW -150,0: DRAW 0,-11
0
  60 LET x=220: GO SUB 1000
  70 PLOT 218,128: DRAW 4,0
  80 LET w$=c$(INT (RND*25)+1)
  90 PRINT AT 8,1;"Cuvint": PRIN
T " de ghicit"
 100 FOR t=1 TO 10
 110 IF w$(t)=" " THEN GO TO 120
 115 NEXT t
 120 LET b=t-1
 130 LET q=0
 150 LET c=1: LET d=0
 160 FOR i=1 TO b
 170 PRINT AT 12,i;"-"
 180 NEXT i
 190 PRINT AT 18,1;"Propune o li
tera"
 200 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 205 LET ind=0
 210 IF r$="" THEN GO TO 200
 220 PRINT AT 20+q,c;r$
 230 LET c=c+1
 240 IF c=30 THEN LET q=1: LET c
=1
 250 FOR i=1 TO b
 260 IF w$(i)=r$ THEN LET v$(i)=
r$: LET ind=1
 270 NEXT i
 280 PRINT AT 11,1;v$
 290 IF v$=w$ THEN GO TO 500
 300 IF ind=1 THEN GO TO 200
 410 IF d=8 THEN BEEP 1,-1: BEEP
 .5,0: BEEP 1,7: BEEP 2,-3: GO T
O 600
 420 LET d=d+1
 430 READ x0,y0,x,y
 440 PLOT x0-20,y0: DRAW x,y
 450 GO TO 200
```

```
 500 FOR i=1 TO 3: FOR j=20 TO 4
0: BEEP .03,RND*20+j: NEXT j: NE
XT i
 510 OVER 1
 520 LET x=220: GO SUB 1000
 530 PLOT 218,128: DRAW 4,0
 540 OVER 0
 550 LET x=126: GO SUB 1000
 560 PLOT 123,129: DRAW 6,0,PI/2
 570 GO TO 800
 600 OVER 1
 620 PLOT 238,65: DRAW -48,0
 630 DRAW 0,-48
 640 PLOT 218,128: DRAW 4,0
 660 PLOT 235,117: DRAW -15,-15:
 DRAW -15,15
 670 OVER 0
 680 PLOT 216,81: DRAW 4,21: DRA
W 4,-21
 690 OVER 1
 700 PLOT 235,66: DRAW -15,15: D
RAW -15,-15
 710 OVER 0
 720 PLOT 216,60: DRAW 4,21: DRA
W 4,-21
 730 PLOT 217,127: DRAW 6,0,-PI/
2
 740 PRINT AT 11,1;w$
 800 PRINT AT 18,1;"Alt joc (d/n
) ? "
 810 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 820 IF r$="d" THEN RESTORE : CL
S : GO TO 10
 830 STOP
1000 CIRCLE x,132,8
1030 PLOT x+4,134: PLOT x-4,134:
 PLOT x,131
1040 PLOT x,123: DRAW 0,-20
1050 PLOT x,101: DRAW 0,-19
1070 PLOT x-15,66: DRAW 15,15: D
RAW 15,-15
1090 PLOT x-15,117: DRAW 15,-15:
 DRAW 15,15
1100 RETURN
2000 DATA 120,65,138,0,184,65,0,
91
2010 DATA 168,65,16,16,184,81,16
,-16
2020 DATA 184,156,68,0,184,140,1
6,16
2030 DATA 204,156,-20,-20,240,15
6,0,-16
```

ROATA NOROCULUI

Programul simulează un « joc mecanic » dintre cele uzuale în locurile cu acestă destinație din stațiunile turistice, anume acele aparate cu manetă, în care se introduc monede și, în funcție de combinația apărută pe un « ecran », se cîștigă sau nu anumite premii. În cazul programului, se consideră că jucătorul (avînd inițial cel mult 99 de lei) poate miza de fiecare dată mai multe monede (între una și suma pe care o are, fără a depăși 99), apoi « trage de manetă » apăsînd o tastă și pierde monedele jucate sau cîștigă un premiu proporțional cu miza, în funcție de combinațiile de cîte trei numere care apar pe ecran. Pentru joc trebuie deci specificată suma de plecare și, la fiecare tragere, miza. Toate aceste numere se consideră formate din cîte două cifre, deci pentru numerele mai mici ca 10 se tastează mai întîi un zero (03 în loc de 3). După fiecare joc se întreabă « Mai joci (d/n)? », comentîndu-se adecvat decizia de renunțare (precum și situația de pierdere a tuturor monedelor). La epuizarea « fondului de joc », programul poate fi reluat.

Descrierea programului

20—120 — pregătirea ecranului.
130 (GOSUB 1000) — desenarea « mașinii de joc ».
140—238 — regulile de joc (pierdere și cîștig).
240—300 — precizarea sumei de plecare (din două cifre, deci prima se înmulțește cu 10 — linia 260); suma este reținută în variabila **fond.**
330—410 — precizarea sumei mizate (tot de două cifre) și verificarea dacă depășește suma disponibilă.
410—450 — alegerea combinației **(x, y, z).**
460—580 — simularea rotirii celor trei discuri ale « mașinii de joc ».
590—610 — scrierea combinației alese **(x, y, z).**
620 — 710 evaluarea combinației (deci a cîștigului), conform regulilor de joc; **m = 0** indică pierdere, **m = 1** indică cîștig.
760 — cîștigul (variabila **cis**) este calculat în funcție de combinație (variabila **m**) și de miză.
770—790 — modificarea fondului curent.
800—910 — comentarii la pierdere (liniile 810—850), opțiune de continuare, dacă mai există fonduri (liniile 860—900) comentariu la renunțare (linia 910).

```
  10 CLS : BORDER 1: PAPER 6: IN
K 2
  20 FOR i=0 TO 31
  30 PRINT AT 0,i;"<CAPS 8>"
  40 PRINT AT 17,i;"<CAPS 8>"
  45 PRINT AT 19,i;"<CAPS 8>"
  50 PRINT AT 21,i;"<CAPS 8>"
  70 NEXT i
  80 FOR i=1 TO 20
  90 PRINT AT i,0;"<CAPS 8>"
  95 IF i>16 THEN GO TO 110
 100 PRINT AT i,17;"<CAPS 8>"
 110 PRINT AT i,31;"<CAPS 8>"
 120 NEXT i
 130 INK 1: GO SUB 1000
 140 PRINT AT 2,18;"REGULI DE JO
C"
 150 PRINT AT 4,18;"1 1 1 = X100
"
 155 PRINT AT 5,18;"MARELE PREMI
U"
 160 PRINT AT 7,18;"crescator=X1
5"
 170 PRINT AT 8,18;"descresc.=X1
2"
 180 PRINT AT 9,18;"n n n = X10"
 190 PRINT AT 10,18;"n n - = X3"
 200 PRINT AT 11,18;"n - n = X2"
 210 PRINT AT 12,18;"1 - - = X1"
 220 PRINT AT 13,18;"- 1 - = X1"
 230 PRINT AT 14,18;"- - 1 = X1"
 235 PRINT AT 15,18;"altceva = "
 238 PRINT AT 16,18;"      pierde
re"
 240 PRINT AT 18,1;"Cit lei ai (
1 - 99) ?"
 250 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 260 IF r$>="0" AND r$<="9" THEN
 LET fond=10*VAL r$: GO TO 280
 270 BEEP 1,-6: GO TO 250
 280 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 290 IF r$>="0" AND r$<="9" THEN
 LET fond=fond+VAL r$: GO TO 310
 300 BEEP 1,-6: GO TO 250
 310 IF fond=0 THEN GO TO 300
 320 PRINT AT 18,1;"Ai in acest
moment ";fond;" lei. "
 330 PRINT AT 20,1;"Citi lei miz
ezi (1 - 99) ?   "
 340 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 350 IF r$>="0" AND r$<="9" THEN
 LET miza=10*VAL r$: GO TO 370
 360 BEEP 1,-6: GO TO 340
 370 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
 380 IF r$>="0" AND r$<="9" THEN
 LET miza=miza+VAL r$: GO TO 400
 390 BEEP 1,-6: GO TO 340
 400 IF miza=0 OR miza>fond THEN
 GO TO 390
 410 PRINT AT 20,1;"Se accepta -
 apasa o tasta !  "
 420 PAUSE 0
 430 LET x=INT (RND*9)+1
 440 LET y=INT (RND*9)+1
 450 LET z=INT (RND*9)+1
 455 INVERSE 1
 460 FOR i=1 TO 4
 470 FOR j=1 TO 9
 480 BEEP .01,i+j-10
 490 PRINT AT 8,6;j
 500 BEEP .01,j+i*3-12
 520 LET k=j+3: IF k>9 THEN LET
k=k-9
 530 PRINT AT 8,9;k
 540 BEEP .01,i+j
 550 LET k=k+2: IF k>9 THEN LET
k=k-9
 560 PRINT AT 8,12;k
 570 NEXT j
 580 NEXT i
 590 PRINT AT 8,6;x: BEEP .03,30
 600 PRINT AT 8,9;y: BEEP .03,30
 610 PRINT AT 8,12;z: BEEP .03,3
0
 615 INVERSE 0
 620 IF x=1 AND y=1 AND z=1 THEN
 LET m=100: GO TO 740
 630 IF x=y-1 AND y=z-1 THEN LET
 m=15: GO TO 740
 640 IF x=y+1 AND y=z+1 THEN LET
 m=12: GO TO 740
 645 IF x=y AND y=z THEN LET m=1
0: GO TO 740
 650 IF x=y THEN LET m=5: GO TO
740
 660 IF x=z THEN LET m=2: GO TO
740
 670 IF x=1 OR y=1 OR z=1 THEN L
ET m=1: GO TO 740
 710 LET m=0
 720 PRINT AT 20,1;"Ai pierdut !
          "
 730 BEEP .3,-6: BEEP .6,-9: BEE
P 1,-20: GO TO 760
 740 PRINT AT 20,1;"Ai cistigat
          "
 750 BEEP .01,16: BEEP .03,29: B
EEP .1,20
 760 LET cis=miza*(m-1)
 765 PAUSE 90
 770 LET fond=fond+cis
 780 IF cis>0 THEN PRINT AT 20,1
4;"(";cis;" lei)"
 790 PRINT AT 18,1;"Ai in acest
moment ";fond;" lei.  "
 800 IF fond>0 THEN GO TO 860
 810 PRINT AT 18,1;"HA HA Ai pie
rdut toti banii !"
 820 PRINT AT 19,1;"Asa-ti trebu
ie daca joci asa  "
 830 PRINT AT 20,1;"ceva SALUT A
lt fraier la rind"
 840 BEEP .2,-3: BEEP .4,2: BEEP
 1,12
 845 PRINT AT 21,1;"(daca exista
,sa apese o tasta)"
```

```
850 PAUSE 0: GO TO 10
860 PRINT AT 20,1;"Doresti sa c
ontinui (d/n) ?  "
870 PAUSE 0: LET r$=INKEY$: BEE
P .1,12
880 IF r$="d" THEN GO TO 330
890 IF r$="n" THEN GO TO 910
900 BEEP 1,-6: GO TO 870
910 PRINT AT 20,1;"Inteleapta h
otarire - bravo !!": STOP
1000 PLOT 36,80: DRAW 78,0: DRAW
 0,55: DRAW -78,0: DRAW 0,-55
1010 PLOT 40,84: DRAW 70,0: DRAW
 0,47: DRAW -70,0: DRAW 0,-47
1020 PLOT 36,80: DRAW -10,6: DRA
W 0,43: DRAW 10,6
1030 PLOT 44,80: DRAW 0,-20: DRA
W 5,0: DRAW 0,20
```

```
1040 PLOT 100,80: DRAW 0,-20: DR
AW 5,0: DRAW 0,20
1050 PLOT 32,83: DRAW 0,-17: DRA
W 5,0: DRAW 0,14
1060 PLOT 87,80: DRAW 0,-14: DRA
W 5,0: DRAW 0,14
1070 PLOT 45,90: DRAW 0,35: DRAW
 13,0: DRAW 0,-35: DRAW -13,0
1080 PLOT 69,90: DRAW 0,35: DRAW
 13,0: DRAW 0,-35: DRAW -13,0
1090 PLOT 93,90: DRAW 0,35: DRAW
 13,0: DRAW 0,-35: DRAW -13,0
1095 INVERSE 1
1100 PRINT AT 8,6;"*"
1110 PRINT AT 8,9;"*"
1120 PRINT AT 8,12;"*"
1125 INVERSE 0
2000 RETURN
```

Ș
E
P
T
I
Ç
Ă

Jocul cunoaște o răspîndire «folclorică» de aceea nu-i mai precizăm regulamentul. Programul — se numește CAMPION — joacă absolut regulamentar contra unui singur partener (nu știe să joace «în trei» sau «în patru»). Cărțile sînt împărțite de fiecare dată de calculator (folosind generatorul de numere aleatoare, deci la întîmplare, «cinstit»), dar partida începe alternativ, un joc el, un joc partenerul. Jucătorul trebuie să-și spună la început numele (apare pe ecran, cu prima literă pîlpîind atunci cînd este la rînd să joace). Pentru indicarea cărții cu care se joacă se folosesc tastele D și N. În dreptul cărților jucătorului se găsește un semn ? pîlpîitor. Dacă se apasă tasta D, este jucată cartea respectivă. Dacă se apasă N, semnul ? se mută mai jos (de la ultima carte, el revine din nou la cea de sus). Tot prin D (= da) și N (= nu) trebuie răspuns la întrebarea «Continui?», pe care calculatorul o formulează în momentul în care taie «pe mîna jucătorului».

De la un joc la altul, punctele se cumulează (desigur, se consideră 16 la 0 atunci cînd nu se ia nici o carte tot jocul).

Cum arătam mai sus, programul joacă «cinstit» fără să se uite în cărțile adversarului. În schimb, el reține toate cărțile care au ieșit din joc, de aceea, atunci cînd joacă (cu precădere atunci cînd joacă primul), alege cu grijă cartea depusă, în funcție de cărțile, folosite deja și de cele pe care le are «în mînă». Pentru că atunci cînd joacă primul joacă bine, el va încerca tot timpul să fie primul, de aceea va tăia uneori fără rost, doar pentru a prelua inițiativa. În momentul în care cărțile sînt epuizate (numărul celor care mai sînt de jucat apare în partea de sus a ecranului), cunoscînd cărțile ieșite și pe cele de care dispune, el cunoaște implicit și cărțile adversarului și ca atare joacă finalul (aproape) perfect. Toate aceste caracteristici fac din programul CAMPION un jucător de șeptică de mare clasă. Desigur, el poate fi bătut, asta depinde și de cărțile distribuite, dar, după un număr mai mare de confruntări, superioritatea sa are toate șansele să iasă la iveală, cîștigînd mai mult d jumătate dintre partide.

Bineînțeles, după fiecare partidă se pune întrebarea «Alt joc (d/n)?» Pentru că programul este foarte lung, conține cîteva comentarii și a fost modificat de un mare număr de ori, renunțăm la descrierea sa (mai ales că nu avem modificări esențiale și relativ simple care i se pot aduce; pentru joc în trei sau în patru este nevoie de un program mult diferit).

```
  5 BORDER 1: PAPER 6: CLS : FO
R i=1 TO 40
  10 PLOT 0,0: DRAW 6*i,170-3*i
  15 PLOT 255,0: DRAW -6*i,170-3
*i
  20 NEXT i
  25 LET meci=0: LET tots=0: LET
 totj=0
  30 DIM n$(8): LET rin=-1
  35 PRINT FLASH 1;AT 6,8;"S E P
 T I C A"
  40 LET v=4: GO SUB 2000
  45 PAUSE 50
  50 PRINT AT 10,0;" Spune-mi nu
mele tau (8 litere) "
  55 INPUT n$: BEEP .05,22: BEEP
 .1,32
  60 PRINT : PRINT " Incepem pe
rind                  "
  65 PAUSE 60
  70 PRINT : PRINT " Esti gata (
d = da) ";n$;" ? "
  75 LET caj=0: LET cas=0
  80 PAUSE 0: LET a$=INKEY$: BEE
P .05,12: BEEP .1,32
  90 IF a$="d" THEN GO TO 130
 100 PRINT : PRINT " Bine, astep
t sa te pregatesti  "
 110 BEEP .2,6: BEEP .6,15: BEEP
 .2,6: BEEP .6,0: BEEP .6,-8: PA
USE 10
 120 IF INKEY$<>"d" THEN GO TO 1
10
 130 IF rin=-1 THEN PRINT : PRIN
T " Incepe CAMPION
   ": GO TO 140
 135 PRINT : PRINT " Incepe ";n$
;"                   "
 140 IF rin=-1 THEN LET a$="C":
GO TO 170
 150 LET a$="j"
 180 DIM c(32): DIM m(8): DIM s(
2,4): DIM f(8)
 200 LET sp=0: LET jp=0: LET cf=
0
 210 FOR i=0 TO 7
 220 FOR j=1 TO 4
 230 LET c(4*i+j)=i+7
 240 NEXT j
 245 NEXT i
 250 PRINT AT 21,0;" Incepem   "
 260 LET v=4: GO SUB 2000
 280 PAUSE 160: LET ce=0: LET cd
=4
 285 CLS
 290 GO SUB 2100
 300 GO SUB 2300
 310 LET mut=1
 320 IF a$="j" THEN GO TO 1200
 325 IF cf=32 THEN GO TO 3500
 330 REM prima mutare calculator
 335 PRINT AT 21,1;"
"
 340 PRINT AT 1,1; FLASH 1;"C"
 345 FOR i=2 TO 8
 350 LET n=0
 360 FOR k=1 TO ce
 370 IF s(1,k)=6+i THEN LET n=n+
1
 380 NEXT k
 390 IF n=0 THEN GO TO 410
 400 IF n+f(i)=4 THEN LET sda=6+
i: GO TO 700
 410 NEXT i
 420 LET max=0
 430 FOR i=1 TO ce
 440 LET n=0
 450 FOR k=i TO ce
 460 IF s(1,i)=s(1,k) AND s(1,i)
<>7 THEN LET n=n+1
 470 NEXT k
 480 IF max<n THEN LET max=n: LE
T sda=s(1,i)
 490 NEXT i
 500 IF max>=3 THEN GO TO 700
 510 IF max<=1 THEN GO TO 620
 520 IF sda<>10 AND sda<>11 THEN
 GO TO 700
 530 IF sda=10 AND f(4)>=1 THEN
GO TO 700
 540 IF sda=11 AND f(5)>=1 THEN
GO TO 700
 545 LET ns=0
 550 FOR i=1 TO ce
 560 IF s(1,i)=7 THEN LET ns=ns+
1
 570 NEXT i
 580 IF ns+f(1)>=3 OR ns>=2 THEN
 GO TO 700
 590 IF ns=1 AND cf<=12 THEN GO
TO 700
 600 LET zar=INT (RND*3)
 610 IF zar=1 THEN GO TO 700
 620 FOR i=1 TO ce
 630 LET sda=s(1,i)
 640 IF sda<>7 AND sda<>10 AND s
da<>11 THEN GO TO 690
 650 NEXT i
 660 FOR i=1 TO ce
 665 IF s(1,i)<>7 THEN LET sda=s
(1,i): GO TO 700
 670 NEXT i
 680 LET sda=7: GO TO 700
 690 GO SUB 2400
 700 REM joaca calculatorul
 800 GO SUB 2500
 810 GO SUB 2700
 815 LET ce=ce-1: GO SUB 2300
 820 IF m(1)=m(mut-1) OR m(mut-1
)=7 THEN GO TO 1080
 830 PRINT AT 21,1;"
          ": PRINT AT 18,4;"Le
 iau"
 835 LET cas=1
 840 LET v=6: GO SUB 2000
 850 PAUSE 60: LET a$="C"
 860 FOR i=1 TO mut-1
 870 IF m(i)=10 OR m(i)=11 THEN
LET sp=sp+1
 880 NEXT i
```

```
 890 LET cd=4-ce
 900 CLS : GO SUB 2300
 910 IF cf=32 THEN GO TO 940
 920 IF cf+2*cd<=32 THEN GO TO 2
90
 930 LET cd=cd-1: GO TO 920
 940 IF ce>0 THEN GO TO 310
 950 CLS : PRINT AT 6,1;"Partida
 s-a terminat"
 960 LET v=5: GO SUB 2000
 970 IF cas=0 THEN LET jp=16
 980 IF caj=0 THEN LET sp=16
 985 LET tots=tots+sp: LET totj=
totj+jp: LET meci=meci+1
 990 PRINT : PRINT ;" Cu scorul:
 CAMPION  = ";sp
1000 PRINT "            ";n$;" =
 ";jp
1010 IF sp>jp THEN LET v=8: GO S
UB 2000: PRINT : PRINT " Invinge
 CAMPION": GO TO 1040
1020 IF sp=jp THEN PRINT : PRINT
 " Egalitate": GO TO 1040
1030 LET v=2: GO SUB 2000: PRINT
 : PRINT " Invinge ";n$
1040 PRINT : PRINT " Jocul nr. =
 ";meci: PRINT " Total: CAMPION
 = ";tots: PRINT "         ";n$;"
 = ";totj: PRINT AT 21,1;"Alt jo
c (d = da) ?"
1045 LET rin=-1*rin
1050 PAUSE 0: LET a$=INKEY$: BEE
P .05,22: BEEP .1,32
1060 IF a$<>"d" THEN CLS : STOP
1070 CLS : GO TO 70
1080 IF ce=0 THEN GO TO 1110
1085 FOR i=1 TO ce
1090 IF m(1)=s(1,i) THEN LET sda
=m(1): GO TO 700
1100 NEXT i
1102 FOR i=1 TO ce
1105 IF s(1,i)=7 THEN LET sda=7:
 GO TO 700
1108 NEXT i
1110 PRINT AT 18,22;"Ia-le": PRI
NT AT 21,1;"                    "
1120 LET caj=1: LET v=3: GO SUB
2000
1130 PAUSE 60: LET a$="j"
1140 FOR i=1 TO mut-1
1150 IF m(i)=10 OR m(i)=11 THEN
LET jp=jp+1
1160 NEXT i
1170 GO TO 890
1200 GO SUB 2700
1205 IF cf=32 THEN GO TO 4020
1210 GO SUB 3000
1215 GO SUB 2500: LET ce=ce-1: G
O SUB 2300
1220 IF m(1)<>sda AND sda<>7 THE
N GO TO 1110
1225 IF ce=0 THEN GO TO 830
1230 PRINT AT 21,1;"Continui (d
= da) ?       "
1240 PAUSE 0: LET d$=INKEY$: BEE
P .05,22: BEEP .1,32
1250 IF d$<>"d" THEN GO TO 830
1255 PRINT AT 21,1;"
          "
1260 PRINT AT 1,23; FLASH 1;n$(1
)
1270 LET i=0
1280 LET i=i+1: IF i=ce+1 THEN L
ET i=1
1290 PRINT AT 5+i*2,25; FLASH 1;
"?"
1300 PAUSE 0: LET d$=INKEY$: BEE
P .05,22: BEEP .1,32
1320 IF d$="n" THEN PRINT AT 5+i
*2,25;" ": GO TO 1280
1330 IF d$<>"d" THEN PRINT AT 21
,1;"Eroare - repeta
 ": BEEP 1,0: PAUSE 40: PRINT AT
 21,1;"                  ": GO TO
 1270
1340 IF s(2,i)<>m(1) AND s(2,i)<
>7 THEN PRINT AT 21,1;"Continuar
e eronata         ": BEEP .6,0
: PRINT AT 5+2*i,25;" ": GO TO 1
230
1350 PRINT AT 21,1;"
          "
1360 LET m(mut)=s(2,i)
1370 PRINT AT 5+i*2,25; FLASH 1;
"<6>"
1380 IF s(2,i)>=10 THEN PRINT AT
 20-mut*2,15; INVERSE 1;s(2,i);
INVERSE 0;" J": GO TO 1390
1385 PRINT AT 20-mut*2,15;"<CAPS
 8>"; INVERSE 1;s(2,i); INVERSE
0;" J"
1390 LET s(2,i)=s(2,ce)
1400 LET f(s(2,i)-6)=f(s(2,i)-6)
+1
1410 LET mut=mut+1: PAUSE 30
1420 PRINT AT 5+i*2,25;"      "
1430 PRINT AT 1,23;n$(1): PAUSE
30
1440 GO TO 1210
2000 REM muzica
2010 FOR i=1 TO 4
2020 BEEP .3/v,INT (RND*7*v)
2025 NEXT i
2030 RETURN
2100 REM se impart carti
2105 PRINT AT 0,15;32-cf;" "
2110 PRINT AT 21,0;" Se impart c
arti   "
2120 FOR j=1 TO 2
2130 FOR i=1 TO cd
2140 LET nr=INT (RND*(32-cf))+1
2145 LET t=1
2150 FOR k=1 TO nr
2170 IF c(t)=0 THEN LET t=t+1: G
O TO 2170
2175 LET t=t+1
2180 NEXT k
2185 LET t=t-1
2190 LET s(j,i+ce)=c(t)
2200 LET c(t)=0
```

```
2210 LET cf=cf+1
2215 PRINT AT 0,15;32-cf;" "
2220 NEXT i
2230 NEXT j
2235 LET ce=ce+cd
2238 PRINT AT 21,1;"
 "
2240 RETURN
2300 REM listeaza
2305 PRINT AT 1,1;"CAMPION": PRI
NT AT 1,23;n$
2310 FOR i=1 TO ce
2340 PRINT AT 5+i*2,3;"(CAPS 88)
"
2350 IF s(2,i)<=9 THEN PRINT AT
5+i*2,27;"(CAPS 8)"; INVERSE 1;s
(2,i): GO TO 2370
2360 PRINT AT 5+i*2,27; INVERSE
1;s(2,i)
2370 NEXT i
2380 FOR i=ce+1 TO 4
2385 PRINT AT 5+i*2,3;"  ": PRIN
T AT 5+i*2,25;"   "
2390 NEXT i
2393 PRINT AT 0,15;32-cf
2395 RETURN
2400 REM cartea slaba cea mai fo
losita
2410 LET max=0
2420 FOR i=1 TO ce
2430 LET ca=s(1,i)
2440 IF ca<>7 AND ca<>10 AND ca<
>11 AND f(ca-6)>max THEN LET max
=f(ca-6): LET sda=ca
2450 NEXT i
2460 RETURN
2500 REM joaca calculatorul
2510 FOR i=1 TO ce
2520 IF sda=s(1,i) THEN GO TO 25
40
2530 NEXT i
2540 PRINT FLASH 1;AT 5+i*2,6;"-
"
2550 PAUSE 60
2560 IF sda>=10 THEN PRINT AT 20
-mut*2,13;"C "; INVERSE 1;sda: G
O TO 2570
2565 PRINT AT 20-mut*2,13;"C (CA
PS 8)"; INVERSE 1;sda
2570 BEEP .05,22: BEEP .1,32: PA
USE 30
2580 PRINT AT 5+i*2,3;"    "
2590 LET f(sda-6)=f(sda-6)+1
2600 LET s(1,i)=s(1,ce): LET m(m
ut)=sda
2610 LET mut=mut+1
2620 PRINT AT 1,1;"C": PAUSE 30
2630 RETURN
2700 REM joaca jucator
2710 PRINT AT 1,23; FLASH 1;n$(1
)
2720 LET i=0
2730 LET i=i+1: IF i=ce+1 THEN L
ET i=1
2740 PRINT AT 5+i*2,25; FLASH 1;
"?"
2750 PAUSE 0: LET d$=INKEY$
2760 BEEP .05,22: BEEP .1,32
2770 IF d$="n" THEN PRINT AT 5+2
*i,25;" ": GO TO 2730
2780 IF d$<>"d" THEN PRINT AT 21
,1;"Eroare - repeta": BEEP 1,0:
PRINT AT 5+i*2,25;" ": PAUSE 40:
 PRINT AT 21,1;"
 ": GO TO 2720
2790 PRINT AT 21,1;"
 "
2800 LET m(mut)=s(2,i)
2810 PRINT AT 5+i*2,25; FLASH 1;
"(6)"
2820 IF s(2,i)>=10 THEN PRINT AT
 20-mut*2,15; INVERSE 1;s(2,i);
INVERSE 0;" J": GO TO 2830
2825 PRINT AT 20-mut*2,15; INVER
SE 1;s(2,i); INVERSE 0;"(CAPS 8)
";" J"
2830 LET s(2,i)=s(2,ce)
2840 LET f(s(2,i)-6)=f(s(2,i)-6)
+1
2850 LET mut=mut+1: PAUSE 30
2860 PRINT AT 5+i*2,25;"   "
2870 PRINT AT 1,23;n$(1): PAUSE
60
2880 RETURN
3000 REM raspunde calculatorul (
prost)
3010 LET ns=0
3020 FOR i=1 TO ce
3030 IF m(1)=s(1,i) THEN LET sda
=s(1,i): GO TO 3200
3040 IF s(1,i)=7 THEN LET ns=ns+
1
3050 NEXT
3060 IF ns=0 THEN GO TO 3120
3070 IF ns>=3 THEN LET sda=7: GO
 TO 3200
3080 IF m(1)<>10 AND m(1)<>11 TH
EN GO TO 3120
3090 IF ns+f(1)>=3 THEN LET sda=
7: GO TO 3200
3100 LET zar=INT (RND*3)
3110 IF zar=1 THEN LET sda=7: GO
 TO 3200
3120 FOR i=1 TO ce
3130 LET ca=s(1,i)
3140 IF ca<>7 AND ca<>10 AND ca<
>11 THEN LET sda=ca: GO TO 3200
3150 NEXT i
3160 FOR i=1 TO ce
3170 IF s(1,i)<>7 THEN LET sda=s
(1,i): GO TO 3200
3180 NEXT i
3190 LET sda=s(1,ce)
3200 RETURN
3500 PRINT AT 1,1; FLASH 1;"C"
3510 LET nsc=0: LET nsj=0
3520 FOR i=1 TO ce
3530 IF s(1,i)=7 THEN LET nsc=ns
c+1
```

```
3535 IF s(2,i)=7 THEN LET nsj=ns
j+1
3540 NEXT i
3550 FOR i=1 TO ce
3560 IF s(1,i)=7 THEN GO TO 3640
3570 LET sda=s(1,i): LET nc=0: L
ET nj=0
3580 FOR j=1 TO ce
3590 IF s(1,j)=sda THEN LET nc=n
c+1
3600 IF s(2,j)=sda THEN LET nj=n
j+1
3610 NEXT j
3620 IF nc+nsc<=nj+nsj THEN GO T
O 3640
3630 LET r=0: GO TO 3750
3640 NEXT i
3650 LET r=1
3660 FOR i=1 TO ce
3670 LET sda=s(1,i)
3690 IF sda<>7 AND sda<>10 AND s
da<>11 THEN GO TO 3750
3700 NEXT i
3710 FOR i=1 TO ce
3720 IF s(1,i)=7 THEN LET sda=7:
 GO TO 3750
3730 NEXT i
3740 LET sda=s(1,ce)
3750 GO SUB 2500
3760 GO SUB 2700
3770 LET ce=ce-1: GO SUB 2300
3780 IF m(1)=m(mut-1) OR m(mut-1
)=7 THEN GO TO 3870
3790 PRINT AT 18,4;"Le iau"
3800 LET cas=1: LET v=6: GO SUB
2000
3810 PAUSE 60: LET a$="C"
3820 FOR i=1 TO mut-1
3830 IF m(i)=10 OR m(i)=11 THEN
LET sp=sp+1
3840 NEXT i
3845 CLS : GO SUB 2300
3850 IF ce>0 THEN LET mut=1: GO
TO 3500
3860 GO TO 950
3870 IF r=0 AND ce>0 THEN GO TO
3960
3880 PRINT AT 18,22;"Ia-le"
3890 LET caj=1: LET v=3: GO SUB
2000
3895 CLS : GO SUB 2300
3900 PAUSE 60: LET a$="j"
3910 FOR i=1 TO mut-1
3920 IF m(i)=10 OR m(i)=11 THEN
LET jp=jp+1
3930 NEXT i
3940 IF ce=0 THEN GO TO 950
3950 LET mut=1: GO TO 1200
3960 FOR i=1 TO ce
3970 IF m(1)=s(1,i) THEN GO TO 3
750
3980 NEXT i
3990 FOR i=1 TO ce
4000 IF s(1,i)=7 THEN LET sda=7:
 GO TO 3750
4010 NEXT i
4020 PRINT AT 1,1; FLASH 1;"C":
LET nsc=0: LET ntc=0: LET ntj=0
4030 FOR i=1 TO ce
4040 IF s(1,i)=7 THEN LET nsc=ns
c+1: GO TO 4055
4050 IF s(1,i)=m(1) THEN LET ntc
=ntc+1
4055 IF i=ce THEN GO TO 4070
4060 IF s(2,i)=m(1) OR s(2,i)=7
THEN LET ntj=ntj+1
4070 NEXT i
4080 IF ntj>=nsc+ntc THEN GO TO
4340
4090 IF ntc>0 THEN LET sda=m(1):
 GO TO 4110
4100 LET sda=7
4110 GO SUB 2500
4120 LET ce=ce-1: GO SUB 2300
4125 IF ce=0 THEN GO TO 3790
4130 PRINT AT 21,1;"Continui (d
= da) ?            "
4140 PAUSE 0: LET a$=INKEY$: BEE
P .05,22: BEEP .08,33: PRINT AT
21,1;"                        "
4150 IF a$<>"d" THEN GO TO 3790
4160 PRINT AT 1,23; FLASH 1;n$(1
)
4170 LET i=0
4180 LET i=i+1: IF i=ce+1 THEN L
ET i=1
4190 PRINT AT 5+i*2,25; FLASH 1:
"?"
4200 PAUSE 0: LET d$=INKEY$
4210 BEEP .05,22: BEEP .08,33
4220 IF d$="n" THEN PRINT AT 5+i
*2,25;" ": GO TO 4180
4230 IF d$<>"d" THEN PRINT AT 21
,1;"Eroare - repeta": BEEP .5,-6
: PAUSE 30: PRINT AT 21,1;"
               ": GO TO 4170
4240 IF s(2,i)<>m(1) AND s(2,i)<
>7 THEN PRINT AT 21,1;"Continuar
e eronata": BEEP .5,-6: GO TO 41
30
4260 LET m(mut)=s(2,i)
4270 PRINT AT 5+i*2,25; FLASH 1;
"<6>"
4280 IF s(2,i)>=10 THEN PRINT AT
 20-mut*2,15; INVERSE 1;s(2,i);
INVERSE 0;" J": GO TO 4290
4285 PRINT AT 20-mut*2,15;"<CAPS
 8>"; INVERSE 1;s(2,i); INVERSE
0;" J"
4290 LET s(2,i)=s(2,ce)
4300 LET mut=mut+1: PAUSE 40
4310 PRINT AT 5+i*2,25;"
      "
4320 PRINT AT 1,23;n$(1): PAUSE
30
4330 GO TO 4020
4340 LET r=1: FOR i=1 TO ce
4342 LET sda=s(1,i)
4345 IF sda=7 OR sda=m(1) OR sda
=10 OR sda=11 THEN GO TO 4355
```

```
4346 IF ce=1 THEN GO TO 4410
4347 FOR j=1 TO ce-1
4349 IF sda=s(2,j) THEN GO TO 43
55
4351 NEXT j
4353 GO TO 4410
4355 NEXT i
4360 FOR i=1 TO ce
4363 LET sda=s(1,i)
4365 IF sda=7 OR sda=m(1) THEN G
O TO 4375
4366 IF ce=1 THEN GO TO 4410
4367 FOR j=1 TO ce-1
4369 IF sda=s(2,j) THEN GO TO 43
75
4370 NEXT j
4372 GO TO 4410
4375 NEXT i
```

```
4380 FOR i=1 TO ce
4383 LET sda=s(1,i)
4385 IF sda<>7 AND sda<>10 AND s
da<>11 THEN GO TO 4410
4387 NEXT i
4390 FOR i=1 TO ce
4392 LET sda=s(1,i)
4395 IF sda<>7 AND sda<>m(1) THE
N GO TO 4410
4397 NEXT i
4400 LET r=0: LET sda=s(1,ce)
4410 GO SUB 2500
4420 LET ce=ce-1: GO SUB 2300
4430 IF m(mut-1)<>m(1) AND m(mut
-1)<>7 THEN GO TO 3880
4440 IF ce=0 THEN GO TO 3790
4450 GO TO 4130
```

Tehnici și artificii pentru programarea jocurilor în limbaj BASIC

Pentru îmbunătățirea jocurilor sau pentru programarea altora, considerăm utilă prezentarea unor tehnici și artificii de programare pe care cititorii le vor putea experimenta, ei înșiși, pe programele prezentate. Scopul este dublu: pe de o parte jocurile vor marca creșteri calitative importante, în special sub aspect estetic iar pe de altă parte se va deschide orizontul spre o cunoaștere mai aprofundată a limbajului **BASIC,** *spre o utilizare avansat a calculatoarelor. Sînt descrise tehnici specifice de programare, cîteva sfaturi utile în vederea evitării unor erori pe care un programator neexperimentat în jocuri le poate comite, o mică colecție de subrutine care se pot utiliza deseori în programare, cîteva « trucuri » de programare etc.*
Ca și jocurile programate, toate aceste tehnici se referă la limbajul BASIC pentru calculatoare compatibile cu tipul **Sinclair ZX Spectrum,** *cu o memorie internă la dispoziția utilizatorului de* **48Ko** *(putîndu-se experimenta pe* **HC, TIM, COBRA, CIP sau JET** *). Multe din tehnicile prezentate fac apel și la cunoștințe mai aprofundate legate de organizarea internă a calculatorului, cum ar fi: instrucțiuni și comenzi* **BASIC** *specifice, variabilele de sistem, organizarea memoriei interne etc. Pentru a veni în sprijinul cititorului, odată cu experimentarea acestor tehnici s-a căutat și prezentarea unora din aspectele mai importante legate de organizarea internă a tipului de calculator amintit.*
*Referindu-ne la artificii de programare (în special cele legate de organizarea internă a calculatorului), ele nu vor putea fi folosite la alte tipuri de calculatoare (***aMIC, PRAE** *etc), tocmai din cauza unei organizări interne diferite. Pentru unele din tehnicile și artificiile prezentate s-au făcut exemplificări chiar pe jocurile prezentate, altele s-au utilizat efectiv în jocuri, existînd, în sfîrșit, și o categorie pentru care invităm pe cititor să le experimenteze singur. Oricum, artificiile prezentate nu au pretenția de a epuiza întreg setul posibil (sîntem siguri că cititorii vor « descoperi » și alte artificii, poate chiar cînd vor experimenta pe cele propuse) iar soluțiile sînt doar niște propuneri susceptibile și ele de a fi îmbunătățite.*
Sperăm totodată, ca prin invitația adresată cititorilor de a experimenta diverse soluții, să stimulăm imaginația și creativitatea acestora, de a găsi noi soluții, originale și spectaculoase.

I. TEHNICI DE PROGRAMARE ŞI SFATURI PENTRU EVITAREA UNOR ERORI ÎN PROGRAMAREA JOCURILOR

1. Utilizarea cu prudenţă a structurilor în care deciziile se iau pe baza comparării a două numere (IF X = Y THEN...), deoarece pot apărea erori datorită modului în care sînt rotunjite numerele (în virgulă mobilă) în interpretorul BASIC:

a) la numerele provenite din scăderi: Pentru a vă convinge rulaţi programul:

```
100 LET X=3.25-3-.25
110 PRINT X
```

Veţi observa că pe ecran se va afişa valoarea 1.1641532E−10 care este foarte mică, dar nu egală cu 0. Dacă valoarea X, astfel calculată, o vom utiliza ulterior într-o linie IF X = 0 THEN..., programul va avea o funcţionare eronată.

Pentru evitarea acestei erori (în funcţionarea logică a programului) sugerăm utilizarea cu precădere a funcţiei valoare absolută (ABS) asupra diferenţei.

b) la numere provenite din ridicări la puteri (pare).

Exemplificare:

```
100 LET A=5*5
110 LET B=5^2
120 PRINT A,B
130 IF A=B THEN GO TO 150
140 PRINT "NU E BINE": STOP
150 PRINT "OK": STOP
```

Veţi observa că se va afişa NU E BINE. La fel, dacă vom utiliza A şi B ulterior într-o linie IF A = B THEN... programul va avea o funcţionare eronată.

Pentru evitarea acestei erori sugerăm utilizarea înmulţirilor repetate (în locul semnului ↑) pentru calcularea valorilor provenite din ridicările la puteri.

c) la numere provenite din împărţiri care nu dau un rezultat exact. Eroarea de rotunjire provenită pe această cale se datorează şi unei greşeli din cadrul sistemului de operare a calculatorului, din care cauză se pierde un (ultim) bit.

Exemplificare:

```
10 INPUT A,B
20 LET A=A/B
30 IF A=0 THEN PRINT "ATINS 0"
: STOP
40 GO TO 20
```

Dacă se vor introduce pentru A şi B valorile 1 şi respectiv 2 programul va intra într-o buclă infinită. Dacă se vor introduce valorile 1 şi 3, după circa 1 sec (82 de cicluri) se va afişa ATINS 0.

Reamintim că în cadrul limbajului BASIC pentru calculatoarele arătate o condiţie care are valoarea 0 este falsă, iar o condiţie care are valoarea diferită de 0 este considerată adevărată.

În mod evident, pot apărea şi erori mari cînd se efectuează calcule numeroase asupra valorilor rezultate din exemplele a), b) şi c), erorile de rotunjire putîndu-se amplifica.

2. Utilizarea ramurilor multiple.

În lipsa unei instrucţiuni GO TO calculat pentru BASIC-ul calculatoarelor avute în vedere (instrucţiuni care optimizează structurile de program), se pot folosi mai multe ramuri de program prin simularea unei instrucţiuni de tipul amintit. Se va folosi o variabilă (A) în funcţie de valoarea căreia programul va « intra » pe o anumită ramură. Simularea se va face printr-o linie GO TO 100×A.

3. Ieşirea dintr-un ciclu FOR-NEXT.

Se recomandă evitarea ieşirii forţate (cu GO TO) dintr-un ciclu FOR-NEXT (pînă cînd variabila indice nu şi-a epuizat toate valorile). Deşi nu este interpretată ca o eroare, este posibil ca acest lucru să conducă la funcţionări eronate ale programului, variabila indice rămînînd la valoarea din momentul ieşirii forţate din ciclu.

Se recomandă, de asemenea, utilizarea ciclurilor prin incrementarea indicelui cu LET faţă de ciclurile FOR-NEXT.

4. Instrucţiunea INPUT.

Se recomandă precedarea instrucţiunii INPUT de un mesaj, deoarece spre deosebire de alte sisteme BASIC (PRAE, TPD JUNIOR etc.), o instrucţiune INPUT

nu produce automat un semn de întrebare pe ecran, atenţionînd utilizatorul asupra faptului că se aşteaptă o introducere. De asemenea, se recomandă ca o instrucţiune INPUT să fie urmată de un PRINT care să afişeze ceea ce a fost introdus. Se atenţionează asupra faptului că acţionarea tastei BREAK nu va avea ca efect oprirea unui program atunci cînd se aşteaptă o introducere (aceasta va fi interpretată de INPUT ca spaţiu). În acest caz, pentru oprirea programului se va acţiona STOP (A+SYMBOL SHIFT).

5. Iniţializarea variabilelor.

Orice variabilă trebuie iniţializată printr-o instrucţiune de asignare sau printr-o instrucţiune DIM înainte de a fi folosită în membrul drept al unei instrucţiuni de asignare. În caz contrar, programul se va întrerupe cu mesajul de eroare corespunzător. Menţionăm că pentru alte sisteme (PRAE, BASIC 80), orice variabilă (neiniţializată), este iniţializată automat la 0, nesemnalîndu-se o eroare. Deşi aparent « drastică » la început, regula privind iniţializarea variabilelor pentru calculatoarele compatibile Sinclair ZX Spectrum este foarte folositoare la depanarea programelor: se va semnala orice variabilă a cărei iniţializare a fost omisă şi, de asemenea, orice variabilă al cărui nume s-a tastat greşit.

6. Generarea liniilor de program similare.

Un mod rapid de a genera mai multe linii de program similare este introducerea primei linii, apoi editarea ei şi modificarea numărului de linie prin facilităţile oferite de editare.

7. Atenţie la folosirea instrucţiunilor: SCREEN$, STR$ şi PAUSE care prezintă unele anomalii (datorate unor greşeli în sistemul BASIC al calculatoarelor Sinclair ZX Spectrum).

Exemplificări:

```
10 PRINT "1234"
20 LET a$=SCREEN$ (0,0)+SCREEN
$ (0,1)
30 PRINT a$
```

La rulare se va afişa 22 în loc de 12.

Pentru evitarea erorilor provenite astfel se recomandă să nu se utilizeze semnul + pentru valorile SCREEN$ (ele să se adauge cîte una).

PRINT "AAA" — "BBB" —STR$ .001

Se va afişa .001.

Pentru evitarea erorilor, se recomandă să nu se utilizeze STR$ la valori cuprinse în intervalul (—1, 1).

```
10 PRINT "ELIBEREAZA TASTA CIN
D SE AUDE BEEP"
20 FOR I=1 TO 500: NEXT I
30 BEEP 1,30
40 PAUSE 0
50 PRINT "SFIRSIT"
```

La rulare se va afişa SFÎRŞIT, deoarece dacă se acţionează o tastă înainte de PAUSE, aceasta este ignorată.

8. Linia pentru salvare-verificare la sfîrşitul programului.

Este indicat ca un joc să aibă încorporată (recomandabil ultima linie de program după un stop) o facilitate de salvare-verificare program, astfel încît utilizatorul să-şi poată salva şi verifica programul după ce l-a introdus în memorie. Linia respectivă va fi: 9999 SAVE "< nume joc >" LINE < număr >: VERIFY ,,< nume joc >". Această tehnică s-a folosit la cîteva din jocurile prezentate ca: VINATOARE, SIMULTAN, DAME etc.

9. Utilizarea tehnicii selectării opţiunii dintr-o listă de opţiuni şi a dialogului între jucător şi calculator în scopul creării unui climat prietenos, a sugerării unui partener plăcut şi inteligent pentru jucător. Astfel, selectarea nivelului de joc, indicarea dorinţei de a cunoaşte sau revedea instrucţiunile jocului, indicarea dorinţei de a continua sau nu jocul sau de a juca un alt joc se vor face de către jucător prin introducerea răspunsului la întrebarea (întrebările) afişată pe ecran. La analizarea răspunsului este recomandabil să se epuizeze toate variantele posibile de răspuns (de exemplu: la întrebarea: Doriţi alt joc? să se analizeze răspunsurile posibile ,,D", ,,d", ,,N", ,,n", ,,Da", ,,da", ,,Nu", alte taste în afară de D,d, N, n etc). Se pot utiliza în acest scop structuri decizionale cu

şiruri de caractere. Mesaje se pot afişa în partea inferioară a ecranului (liniile 22 şi 23), folosind PRINT #0 şi PRINT #1.

10. Utilizarea lui INKEY$.
Este recomandată utilizarea lui INKEY$ mai ales în jocurile de îndemînare şi reflexe, nemafiind necesară în acest caz şi acţionarea lui ENTER (CR).
De asemenea, este recomandată utilizarea lui INKEY$ şi în structuri decizionale pentru selectarea unei ramuri a programului.

11. Sfîrşit de joc.
Prezentăm un exemplu de sfîrşit de joc posibil, care foloseşte INKEY$ pentru selectarea unei ramuri, afişarea de mesaje în partea inferioară a ecranului, analiza răspunsului, dialog politicos etc. De asemenea, se observă folosirea formei INPUT '' '' cu scopul de a se şterge mesajul afişat în partea inferioară a ecranului, fără a se renunţa la informaţiile afişate pe ecran (cum s-ar fi întîmplat la folosirea lui CLS). O tehnică similară se utilizează în jocul ANIMALE.

```
  10 REM Inceput joc
  20 PRINT "Inceput joc"
1000 REM Secventa de sfirsit
1010 LET R$=INKEY$
1020 PRINT #1;"INCA UN JOC?"
1030 BEEP 0.5,2: PAUSE 200
1040 LET R$=INKEY$
1050 IF R$="" THEN GO TO 10
1060 INPUT ""
1070 IF R$="d" OR R$="D" THEN CL
S : GO TO 10
1080 IF R$<>"n" AND R$<>"N" THEN
 GO TO 1000
1090 REM SFIRSIT JOC EVENTUAL CU
 SECVENTA SONORA
1100 CLS : PRINT AT 8,8;"LA REVE
DERE !": STOP
```

II. COLECŢIE DE SUBRUTINE UTILIZABILE ÎN PROGRAMAREA JOCURILOR

1. Subrutine pentru afişarea mesajelor

Mesajele se pot afişa în mod obişnuit cu ajutorul instrucţiunilor PRINT şi PRINT AT. Este indicat ca mesajele pentru jucător să fie afişate în partea de jos a ecranului (eventual cu PRINT #0 şi PRINT #1). Apariţia unui mesaj în alt mod decît uzual poate face această parte a programului mult mai atractivă.
În exemplul dat (care se poate folosi chiar în forma prezentată la jocul Cursa cu obstacole), litera care va trebui afişată este plimbată pe ecran şi adusă la locul potrivit odată cu un sunet muzical. Subrutina pentru afişarea mesajului începe la linia 1400, iar în variabila şir de caractere a$ se memorează mesajul care va fi afişat. De remarcat pentru alte programe că mesajul afişat poate fi oricît de lung (chiar mai lung decît un rînd — 32 caractere), în acest caz, el afişîndu-se pe mai multe rînduri.

```
  10 LET a$="  OBSTACOLE-JOC DE
INDEMINARE"
  15 LET linie=0
  17 GO SUB 1400
  18 REM RESTUL PROGRAMULUI. DAC
A SE FOLOSESTE NUMAI SUBRUTINA P
ENTRU DEMONSTRATIE SE VA ADAUGA
LINIA   19 STOP
1400 FOR x=1 TO LEN a$
1410 PRINT AT linie,0;a$( TO x)
1420 BEEP .05,CODE a$(x)/4
1430 IF x=LEN a$ THEN BEEP .1,30
 : BEEP .1,20
1440 NEXT x
1450 LET linie=linie+1
1460 RETURN
```

Linia 1420, care realizează un sunet specific pentru fiecare caracter al mesajului (în funcţie de codul caracterului respectiv), se poate înlocui cu altă linie (mai simplă) care realizează acelaşi sunet pentru fiecare caracter: **1420 BEEP .05.0**
Următoarea subrutină realizează aproximativ acelaşi lucru, dar pe ecran nu se va plimba cîte un caracter, ci tot mesajul odată, care va fi adus în locul stabilit. În acest caz, mesajul (de fapt ce se memorează în variabila **a$**) nu va trebui să depăşească 32 de caractere. La fel ca şi prima subrutină, cea de a doua se poate folosi în jocul **Cursa cu obstacole.**

```
10 LET a$=" OBSTACOLE -.JOC DE
INDEMINARE  "
 15 LET linie=0
 17 GO SUB 1400
 18 REM RESTUL PROGRAMULUI. DA
CA SE FOLOSESTE NUMAI SUBRUTINA
PENTRU DEMONSTRATIE SE VA ADAUGA
 LINIA   19 STOP

1400 LET b$="
                                  "
1410 LET b$=b$+a$
1420 FOR x=1 TO LEN b$-31
1430 PRINT AT linie,0;b$(x TO x+
31)
1440 NEXT x
1450 RETURN
```

Iată și o subrutină care afișează mesaje cu lungimea maximă de un rînd (32 caractere), prin așa numitul efect de șenilă: caracterele apar unul după altul din partea dreaptă a ecranului și dispar în partea stîngă ca și cum ar fi legate de o șenilă care le readuce din nou prin partea dreaptă pe ecran:

```
 10 LET a$=" OBSTACOLE - JOC DE
 INDEMINARE "
  15 LET LINIE=0
  17 GO SUB 1400
  18 REM RESTUL PROGRAMULUI.DACA
 SE FOLOSESTE NUMAI SUBRUTINA PE
NTRU DEMONSTRATIE SE VA ADAUGA
LINIA    19 STOP
1400 LET a$=a$(2 TO 32)+a$(1)
1410 PRINT AT linie,0;a$
1420 BEEP .1,20
1430 IF INKEY$="" THEN GO TO 140
0
1440 RETURN
```

Pentru ieșirea din subrutina de afișare a titlului va trebui să se acționeze orice tastă. Linia 1420 se poate înlocui cu **BEEP .1,CODE a$/4** dacă se dorește ca apariția unui caracter pe ecran să fie însoțită de un sunet specific sau cu **PAUSE 8** dacă se dorește ca afișarea titlului să se facă în liniște.

Uneori apare necesitatea atenționării asupra unor mesaje importante. În afară de o atenționare sonoră a acestor mesaje (sau a titlurilor jocurilor) se poate realiza și una grafică prin: subliniere, videoinversare sau clipire (FLASH).

Prezentăm o subrutină cu ajutorul căreia se poate sublinia un mesaj afișat pe ecran fără a se ocupa linia de caracter de sub mesaj. Se folosesc în acest scop instrucțiunile **OVER** și caracterele de control **CHR$ 8** (cursor înapoi) și **CHR$ 95 (—)**. Subrutina se poate utiliza efectiv în jocul **Traversarea străzii.**

```
 15 LET x$="Traversare"
 18 FOR i=1 TO LEN x$
 20 PRINT OVER 1;AT 1,6+i:x$(i)
:CHR$ 8;CHR$ 95
 21 NEXT i
```

Pentru sublinierea ulterioară a afișării mesajului:

```
 15 PRINT AT 1,6;"Traversare"
 18 FOR i=0 TO 9
 20 PRINT OVER 1;AT 1,6+i;CHR$
95
 21 NEXT i
```

Se poate realiza și o subliniere a mesajului prin videoinversare utilizîndu-se caracterul **CHR$ 143** (videoinvers). În acest scop se înlocuiește în linia **30 CHR$ 95** cu **CHR$ 143.**

La oricare din cele două variante, pentru atenționare sonoră se pot face următoarele modificări:

linia 20 se va transforma în linia 19, iar noua linie 20 va putea fi:

20 PAUSE 3

sau

20 BEEP .05,15+i

2. Subrutine pentru grafică

Instrucțiunile pentru culori și grafică **(INK, PAPER, BORDER, BRIGHT, INVERSE** și **OVER)** sînt foarte puternice: ele se pot exprima sub formă parametrică nu numai prin valori (întregi) dar și prin expresii și variabile și, de asemenea, ele pot fi obținute prin intermediul cuvîntului BASIC **CHR$** (urmat de un număr de la 16 la 21).

Recomandăm ca inițial un joc să fie rea-

lizat în alb/negru și a se verifica funcționarea lui, organizarea înlănțuirii rutinelor, ameliorarea rapidității etc. și după aceea, în cea de a doua etapă, să se introducă culorile, avînd în vedere întreg ansamblul instrucțiunilor și subrutinelor de grafică. Atribuirea anumitor valori pentru culori (1 pentru albastru, 2 pentru roșu etc) nu este întîmplătoare, între ele existînd legături bazate pe principiul combinării a trei culori de bază care sînt: albastru, verde și roșu. Rezultă că pentru calculatoarele studiate există 3 culori de bază, 3 culori rezultate din combinația acestora, plus o singură culoare (alb) rezultată prin suprapunerea celor 3 culori de bază (negrul, a 8-a culoare se consideră a reprezenta absența culorii).

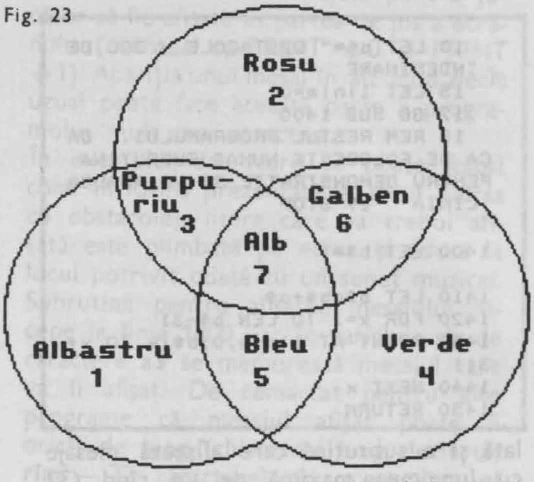
Amestecare culori

Următorul program ilustrează principiul combinării (amestecării) culorilor utilizînd funcțiile logice. Tehnicile de obținere a culorilor prin amestec sînt utile în unele jocuri în care modificarea culorii poate corespunde unei anumite faze (secvențe).

```
10 FOR I=0 TO 21
20 FOR J=0 TO 31
30 LET albastru=(I)1 AND I(15
AND J)7 AND J(23)
40 LET rosu=2*(I)5 AND I(20 AN
D J)2 AND J(14)
50 LET verde=4*(I)11 AND I(18
AND J)10 AND J(30)
60 LET culoare=albastru+verde+
rosu
70 PRINT PAPER culoare;AT I,J;
" "
80 NEXT J: NEXT I
```

Logica combinării culorilor în program se poate urmări în fig. 23.

Fig. 23



36 de culori

Într-adevăr, utilizatorul nu este limitat la cele 8 culori de bază ale calculatorului. Recurgînd la un artificiu foarte simplu, el va dispune de un număr de culori mult mai mare prin amestecarea culorilor. Este suficient în acest scop să se definească în zona rezervată caracterelor definite de utilizator (**UDG**) unul sau mai multe caractere reprezentînd modele mai mult sau mai puțin dense. Astfel, ceea ce pe un display alb/negru va fi reprezentat printr-un gri, pe unul color va deveni un amestec de culori, ochiul avînd iluzia unei culori intermediare între aceea a hîrtiei (**PAPER**) și aceea a cernelii (**INK**). Bineînțeles, efectul va fi satisfăcător dacă cele două culori de amestec nu vor fi foarte contrastante: roșu și galben, verde și albastru etc.

Subrutina prezentată utilizează proprietatea expusă. După ce s-a definit un caracter gri uniform (linia 2000), se va apela succesiunea de culori a curcubeului (linia 4010) și se va realiza afișarea alternativă a unei linii de culoare normală cu una de o culoare amestecată cu aceeași culoare și cu culoarea care urmează.

```
2000 DATA 170,85,170,85,170,85,1
70,85
2010 FOR I=0 TO 7: READ R: POKE
(65368+I),R: NEXT I: STOP
4000 REM Culori amestecate
4010 DATA 1,5,5,4,4,6,6,2,2,3
4015 FOR C=0 TO 4: READ I,J
4020 PRINT PAPER I;"
                                "
4030 PRINT PAPER I; INK J;"▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
4040 NEXT C
8890 STOP
9000 INK 0: PAPER 7: BORDER 7: C
LS
```

Efectul pe ecran va fi acela al unei succesiuni de culori mult mai nuanțate. Pentru utilizare se va introduce comanda

RUN și se va obține caracterul grafic utilizator corespunzător tastei **A.** Apoi se va introduce programul începînd cu linia 4000 și după aceea cu **GO TO 4000** se vor obține cele 10 culori (amestecate) ale curcubeului.

Video-inversarea unei imagini afișate

După ce s-a sfîrșit pe ecran un desen sau un text, pentru a da răgazul necesar analizării imaginii este de efect trecerea acesteia pe revers prin baleierea ecranului cu o linie de grosimea unui pixel, alegînd unul din următoarele moduri:
a) de jos în sus:

```
100 OVER 1
110 FOR I=0 TO 175
120 PLOT 0,I: DRAW 255,0
130 NEXT I: OVER 0
```

b) de sus în jos: programul anterior în care se modifică linia 110:

```
110 FOR I = 175 TO 0 STEP −1.
```

c) de la stînga la dreapta:

```
100 OVER 1
110 FOR I=0 TO 255
120 PLOT I,0: DRAW 0,175
130 NEXT I: OVER 0
```

d) de la dreapta la stînga: programul anterior în care se modifică linia 110:

```
110 FOR I = 255 TO 0 STEP −1
```

JECO

Deplasarea transparentă a unui mobil

Metoda clasică de deplasare a unui mobil (afișarea mobilului pe noua sa poziție și ștergerea lui de pe poziția anterioară) prezintă inconvenientul că distruge iremediabil conținutul pozițiilor traversate de mobil. Aceasta are ca efect lăsarea unei « urme» pe traiectoria mobilului și distrugerea decorului desenat pe ecran. Cîteodată însă, este indispensabilă păstrarea anumitor desene pe ecran.
Instrucțiunea **PRINT OVER 1** permite afișarea printr-o operație de tipul SAU EXCLUSIV a două caractere avînd aceeași poziție pe ecran, realizîndu-se astfel o deplasare « **transparentă**» a unui mobil. Mobilul se va suprapune în decorul pe care îl traversează, apoi va dispare lăsînd decorul intact grație proprietății de involuție a lui SAU EXCLUSIV.
Deci, atenție! Operația SAU EXCLUSIV este involutivă, aceasta însemnînd că repetată de două ori are un efect nul.
Pe baza acestei tehnici se va realiza ștergerea mobilului de pe poziția anterioară.
Instrucțiunile PRINT OVER 1 nu păstrează atributele (culoarea, clipirea, sclipirea) pozițiilor traversate. Din fericire se pot utiliza instrucțiunile complementare care permit obținerea unei transparențe totale la nivel de atribute: **PAPER 8; INK 8; FLASH 8; BRIGHT 8.**

Folosirea caracterelor grafice definite de utilizator

Crearea de noi caractere grafice reprezintă o facilitate importantă în special în realizarea de jocuri. Această facilitate este foarte des folosită în programele prezentate, atît pentru cele de îndemînare și reflexe (PING, BILI, STELE, TRAVERSARE, OBSTACOLE etc), cît și pentru cele logice, de strategie sau cu caracter didactic (BROSC, NIM, DIPO etc). Atenționăm că pentru crearea de modele de desene care ulterior vor fi animate («spiriduși» de dimensiuni mai mari, decît un caracter), se poate utiliza aceeași tehnică pentru alăturarea mai multor caractere grafice definite de utilizator. De obicei se folosește unirea mai multor caractere grafice definite obținîndu-se un model în formă de pătrat sau dreptunghi. Pentru cei care utilizează des crearea de caractere grafice indicăm realizarea unui program unealtă ajutător, prin intermediul căruia caracterele grafice să se genereze mult mai simplu (editor grafic de caractere). Practic se va simula grafic grila de 8×8, fiind de asemenea necesar ca să existe posibilitatea plimbării pe căsuțele grilei a caracterului grafic «■» (de exemplu prin intermediul tastelor care reprezintă săgețile stînga, dreapta, sus și jos), precum și posibilitatea marcării unei căsuțe cu acest caracter. Bineînțeles, cînd caracterul a fost creat, el se va putea «memora» prin salvarea sa pe caseta magnetică. Utilizînd această tehnică se poate crea și un nou set de caractere pentru calculator care va fi memorat și apelat la nevoie dintr-o anumită zonă a memoriei calculatorului (vezi mai departe utilizarea variabilelor de sistem).

Subrutină pentru mărirea dimensiunilor caracterelor afișate

Caracterele sînt memorate în memoria ROM începînd cu adresa 15616 prin secvențe de 8 octeți. Numărul binar corespunzător fiecărui octet va fi, în acest caz, un șir de **0** și de **1** reprezentînd o parte a modelului caracterului. Acesta este înscris într-o grilă de 8×8 puncte, în care orice **1** va reprezenta un punct (aprins).

Programul prezentat ilustrează principiul de memorare a codurilor corespunzătoare unui caracter și realizează în final o afișare a unui caracter pe o matrice de 8×8 celule caractere. Subrutina poate fi folosită în jocuri atunci cînd se dorește apariția pe ecran a unui caracter de dimensiuni mari (sau a unui mesaj foarte important, deci de dimesniuni mari) și nu se cunoaște dinainte care va fi caracterul sau mesajul respectiv. Pentru a realiza acest lucru, programul găsește în memorie secvența de numere care codifică caracterul a cărei afișare apare ca necesar a fi mărită. În final se va desena forma mărită a caracterului, înscriindu-se în matricea de 8×8 cifre binare cîte un caracter grafic «■» pe locul celulei caracter corespunzătoare pentru cifra binară **1**, lăsîndu-se neschimbate celulele caracter corespunzătoare pentru cifre binare **0**. O tehnică similară s-a propus și pentru realizarea editorului grafic pentru generarea de caractere.

În programul prezentat pentru identificarea locului din memorie unde este memorat caracterul care se dorește a se afișa sub formă mărită, se utilizează următorul algoritm: cunoscîndu-se adresa de memorie de unde încep să se memoreze caracterele (15616—256=15360), adresa

oricărui caracter se va găsi cu formula 15360 + 8 × codul caracterului.
Setul de caractere memorat începe cu simbolul SPACE (codul 32) și se termină cu simbolul © (cod 127). Adresa la care este memorat caracterul fiind identificată, se vor citi și se vor afișa cele 8 adrese consecutive corespunzătoare celor 8 octeți, una cîte una (cu PEEK) și se vor afișa în continuare numerele zecimale precum și formele binare corespunzătoare fiecărui octet. Se va putea obține modelul mărit al oricărui caracter sau simbol (de la codul 32 pînă la 127). Liniile de instrucțiuni 110 și 130 servesc la conversia numărului zecimal L în forma sa binară (se fac împărțiri succesive la 2, iar resturile se citesc și se afișează în ordine normală, nu inversă, deoarece în cazul de față, numărul binar se citește și se scrie de la dreapta spre stînga conform liniei 80). Cu acest principiu se pot reprezenta caractere de orice mărime cu condiția să nu se depășească limitele ecranului. Dar iată rutina:

```
  10 LET A=15360
  20 INPUT "INTRODUCETI UN CARAC
TER ";A$
  30 CLS
  40 LET D=A+8*CODE A$
  50 FOR X=0 TO 7
  60 LET L=PEEK (D+X)
  70 PRINT D+X:TAB 6:L
  80 FOR Z=18 TO 11 STEP -1
  90 PRINT AT X,Z;L-2*INT (L/Z)
 100 IF L-2*INT (L/2)=1 THEN PRI
NT AT X,Z+10;" "
 110 LET L=INT (L/2)
 120 NEXT Z
 130 NEXT X
```

Iată și o altă subrutină care, funcționînd pe baza acelorași principii, va facilita o afișare de dimensiuni mari a unor mesaje pe 3 rînduri, fiecare rînd conținînd 4 caractere (deci, un mesaj de 12 caractere):

```
  10 DIM A(4)
  20 INPUT "INTRODUCETI MESAJUL
(12 CARACTERE)";A$
  30 CLS
  40 FOR F=1 TO 3
  50 FOR L=1 TO 4
  60 LET A(L)=CODE A$(L)
  70 NEXT L
  80 LET A$=A$(5 TO )
  90 FOR B=1 TO 7
 100 FOR C=1 TO 4
 110 LET X=PEEK (15360+A(C)*8+B)
 120 FOR V=0 TO 7
 130 IF X<128 THEN GO TO 170
 140 PRINT " ";
 150 LET X=X-128
 160 GO TO 180
 170 PRINT " ";
 180 LET X=X*2
 190 NEXT V
 200 NEXT C
 210 NEXT B
 220 NEXT F
```

În locul subrutinelor descrise (destul de lungi), în vederea afișării unor mesaje cu caractere mărite se poate folosi un artificiu, care, utilizînd o locație de memorie din zona variabilelor de sistem (și anume **23681)**, realizează afișarea cu caractere mărite a unor texte de maxim un rînd (32 de caractere).
Mărirea caracterelor se face, în acest caz, în primul rînd în înălțimea lor, caracterele rezultate fiind înguste (pentru a se menține o afișare de 32 de caractere pe un rînd).
De exemplu, înlocuiți linia 20 a jocului **Vrăjitorul Portocaliu** cu următoarea linie:

```
  20 FOR I=64 TO 71: POKE 23681,
 I: LPRINT "VRAJITORUL PORTOCALIU
 ": NEXT I
```

Veți observa cum textul dintre ghilimele va apărea afișat cu litere mărite în partea superioară a ecaranului.
De remarcat că artificiul se poate realiza pe 3 zone ale ecranului:
— partea superioară a ecranului, cum a fost în exemplul de față, pentru un indice cuprins între valorile 64 și 71;
— partea de mijloc a ecranului, pentru un indice cuprins între valorile 72 și 79;
— partea inferioară a ecranului, pentru un indice cuprins între valorile 80 și 87.
În acest fel, repetînd de 3 ori procedeul pentru cele 3 părți ale ecranului, se poate realiza afișarea mărită a unui text de 3 linii a cîte 32 de caractere.
Artificiul admite și forma cu **TAB** după **LPRINT** pentru a evita utilizarea excesivă

a blancurilor atunci cînd se dorește să se poziționeze textul din partea dreaptă a ecranului.

3. Subrutine sonore

Adăugarea de mici secvențe sonore are ca efect creșterea calitativă a jocurilor. Acest lucru apare esențial, de exemplu, în cazul jocurilor cu caracter didactic, în care este recomandat să se atenționeze sonor atît un răspuns pozitiv cît și negativ și, mai ales, în cazul jocurilor de îndemînare și reflexe în care mișcările obiectelor sau acțiunile personajelor însoțite de efecte sonore ajută jucătorul și simulează mai bine un joc sau fapt real.

Recomandăm de asemenea utilizarea judicioasă a **BEEP**-ului în cicluri, pentru obținerea de efecte muzicale mai puțin monotone și liniare față de succesiunea plată a repetării de mai multe ori a instrucțiunii. Se indică modificarea pasului de incrementare, astfel încît să depindă de o variabilă. Aceste tehnici permit producerea de sunete mai apropiate de cele reale, mai ales ținînd seama de faptul că, pentru calculatoarele avute în vedere, nu se poate produce decît un sunet odată și fără armonice.

Următoarele exemple demonstrează cîteva din tehnicile utilizate în acest scop, reprezentînd subrutine sonore care se pot folosi în jocuri:

```
10 REM RIS IRONIC
20 FOR K=0 TO 1000
30 LET C=COS K+0.5
40 BEEP 0.1,-10*C+1: BEEP 0.1,
-20*C+1: BEEP 0.1,-30*C+1
50 NEXT K
```

```
10 REM ELICOPTER
20 FOR K=1 TO 1000 STEP 10
30 BEEP 0.001,ABS (SIN (10*K)+
60)
40 NEXT K
```

```
10 REM TREN
20 FOR K=0 TO 1000
30 BEEP 0.05,-2: BEEP 0.05,-2
40 BEEP 0.5,-10: BEEP 0.2,-60:
BEEP 0.2,66
50 NEXT K
```

Se va nota că în primele două exempe s-au utilizat funcții circulare (cosinus și sinus) în scopul obținerii unor efecte interesante.

În subrutinele sonore se va avea în vedere evitarea valorilor corespunzătoare pentru notele introduse, astfel încît, acestea să nu depășească limitele permise pentru înălțime (frecvență) și să poată fi practic auzite. Sunetele foarte joase (care se aud ca niște păcănituri pot fi prelungite pentru a deveni mai naturale prin modificarea conținutului adresei de memorie **23609** (vezi artificii de program prin utilizarea variabilelor de sistem).

Tehnicile descrise au fost utilizate în majoritatea jocurilor prezentate, inclusiv cea privitoare la modificarea conținutului adresei de memorie 23609.

Pentru producerea sunetelor sau zgomotelor se pot folosi și instrucțiunile pentru porturi, portul cu numărul **254** referindu-se la difuzor. Modificați, de exemplu, linia 300 a jocului VRĂJITORUL PORTOCALIU astfel încît ea să devină:

```
300 OUT 254,10: OUT 254,20: LET
ink=INT (RND*8): IF ink=6 THEN
GO TO 300
```

la rularea programului veți observa că acum căderea fulgerului va fi însoțită de zgomote.

ARTIFICII PENTRU PROGRAME

1. TITLU LA LISTARE ŞI PROTEJAREA UNOR LINII DE PROGRAM

Pentru realizarea acestui artificiu sînt necesare cunoştinţe referitoare la organizarea memoriei interne. Începutul unui program BASIC (informaţii referitoare la prima linie de program) se regăseşte în memorie de la adresa **23755**. Citirea acestor informaţii din memorie se face ţinînd seama de codificarea fiecărei linii de program BASIC, şi anume: pe primii doi octeţi (23755 şi 23756) se reprezintă numărul liniei respective (primul octet este cel mai semnificativ), iar următorii doi octeţi (23757 şi 23758) codifică lungimea liniei respective, adică numărul de caractere (al doilea octet este cel mai semnificativ). Pentru codificarea lungimii liniei de program, un caracter este reprezentat pe un octet, la fel ca şi un cuvînt cheie care se reprezintă tot pe un octet. O constantă numerică este urmată de forma sa binară, utilizîndu-se în acest scop caracterul number (cod 14), urmat de octeţii care codifică numărul în baza 2. O variabilă are un format diferit în funcţie de natura ei.

Începînd cu adresa 23759 se regăseşte textul liniei respective (în 23760 va fi primul caracter după cuvîntul cheie cu care debutează orice linie de program). Pentru a exemplifica cele expuse şi a obţine titlul la listare şi protejarea primei linii de program, adăugaţi următoarele linii jocului VRĂJITORUL PORTOCALIU:

După cum se poate observa, în linia 1, după REM, s-au lăsat 11 spaţii (spaces), începîndu-se comentariul de la poziţia 12. În linia 3, s-a realizat introducerea în primii 8 octeţi, corespunzători textului primei linii de program, a caracterului de control, codificat cu 8 (cursor înapoi), astfel încît la listarea programului (după rularea sa), linia 1 nu va mai fi afişată normal, ci ca un titlu (centrat) de program, fără număr de linie.

În linia 5, s-a introdus în locaţiile de memorie 23755 şi 23756 valoarea 0, astfel încît, după rularea programului, linia 1 se va transforma în linia 0. În plus, această linie va fi în continuare protejată, nemaiputînd fi nici ştearsă şi nici editată.

Protecţia asupra liniei 1 se va putea înlătura prin: POKE 23755, INT (NNNN/256): POKE 23756, NNNN−256*INT (NNNN/256) după care numărul primei linii revine la cel iniţial (NNNN).

În cazul nostru, numărul iniţial fiind 1, va fi suficient să se comande: POKE 23756,1

2. UTILIZAREA CARACTERELOR DE CONTROL

Uneori, o **protejare eficientă a listării liniilor de program** poate fi realizată numai prin simpla utilizare a caracterelor de control în cadrul unei instrucţiuni PRINT, prin modificarea culorii cernelii (**INK**) sau a fondului (**PAPER**), astfel încît, culoarea cernelii să fie aceeaşi cu cea a fondului.

Amintim că pentru modificarea culorii fondului (**PAPER**), se trece în modul extins (**E**) şi apoi se acţionează tastă nu-

merică corespunzătoare culorii dorite, obținîndu-se în acest mod modificarea culorii pe toată linia respectivă precum și pe următoarele. Pentru modificarea culorii cernelii (**INK**) se trece în modul extins (**E**) și apoi se acționează tasta numerică corespunzătoare culorii dorite împreună cu tasta **CS**. Se pot obține astfel caractere de culoare albă pe un fond alb.

Pentru exemplificare, în jocul VRĂJITORUL PORTOCALIU puteți introduce linia **2 PRINT AT 1,5; „VRĂJITORUL PORTOCALIU"** iar după primele ghilimele se va introduce caracterul de control care va determina modificarea culorii negre cu care se afișează caracterele, în culoarea albă. Acest lucru se obține după poziționarea cursorului după ghilimele trecînd în modul extins (**E**) și apoi acționînd tasta **7** (corespunzătoare culorii albe) împreună cu **CS**. În acest caz la listare va apare numai:

VRĂJITORUL PORTOCALIU
3 PRINT ''

Bineînțeles că în acest caz linia 20 se va șterge nemaifiind necesară.

La jocul **ROBAC** s-a realizat o protejare a listării programului prin simpla folosire a culorilor pentru fond (PAPER) și pentru cerneală (INK) fără a se mai utiliza nici măcar caractere de control pentru mascarea acestui artificiu. Astfel, după rularea programului, dacă veți dori listarea sa, nu veți reuși, culoarea fondului fiind aceeași cu culoarea cernelii. Pentru a putea realiza listarea pentru acest program va trebui restabilită în prealabil culoarea neagră a cernelii: **INK 0: LIST.**

Caracterele de control care modifică culorile pot fi anulate cu **DELETE** (**CS** și **0**), dar trebuie ținut cont de faptul că pentru orice caracter de control introdus este necesară acționarea de două ori a tastelor pentru DELETE.

Caracterele de control se pot utiliza nu numai în vederea protejării listării, ci și în vederea **protejării unor mesaje** din jocuri, conținute în linii de program cu PRINT. Astfel, introducerea prin texte a unor caractere de control, de exemplu: **TRUE VIDEO** (**CS** și **3**), **INVERSE VIDEO** (**CS** și **4**) și apoi **TRUE VIDEO** (pentru revenirea la modul normal de afișare), culoarea fondului albă (7 în modul E), dacă atunci se introduce linia, culoarea fondului este albă etc., toate acestea pot produce dificultăți atunci cînd se editează linia care le conține, mai ales dacă linia abundă cu caracterele de control menționate. De remarcat este faptul că linia apare « curată » la listare, aceste caractere de control fiind invizibile la listare. La listările la imprimantă (în listinguri) caracterele de control apar sub forma unor semne de întrebare.

Pentru editarea liniei, fiecare caracter de control va putea fi înlăturat prin acționarea de două ori a tastei DELETE (prima oară apare un semn de întrebare, în locul în care fusese inserat caracterul de control, iar acesta va fi înlăturat prin a doua acționare a tastei DELETE).

Folosind caracterele de control pentru deplasarea cursorului se poate realiza și modificarea numelui unui joc salvat pe caseta magnetică sau, cu alte cuvinte, se poate realiza un **heder (antet) « fals ».** Numele unui program BASIC este o expresie de tip șir de caractere de maxim 10 caractere și el reprezintă numele sub care a fost salvat programul pe caseta magnetică. Atunci cînd la o operație de încărcare a unui program BASIC se identifică un anume program, pe ecran se afișează mesajul:

Program: < nume program >

Dacă la comanda de salvare a jocului NIM vom introduce:

SAVE CHRS 8+ ''el: NIM''

atunci la încercarea de a încărca alt joc sau chiar pe acesta va apărea mesajul:

Programul: NIM

deoarece cele două caractere de control introduse în numele programului vor determina deplasarea cursorului de două ori spre stînga. Se pot crea hedere « false » și mascînd complet mesajul « Program » și de asemenea hedere cu « clipire ». Pentru primul caz, vom lua de exemplu jocul PING, căruia îi vom adăuga următoarele linii:

```
   2 LET a$=""
   3 LET a$=a$+CHR$ 22+CHR$ 1+CH
R$ 0+"PING"+CHR$ 23
   4 REM RESTUL PROGRAMULUI
1100 SAVE a$
```

Cu **GO TO 1100** vom realiza o salvare a jocului astfel încît la încărcarea ulterioară va apărea numai «PING».

Pentru cel de-al doilea caz, restricția de 10 caractere în titlu este foarte puternică, nemairămînînd loc decît pentru denumiri foarte scurte (de 3 sau mai puține caractere). În schimb, se pot utiliza codurile pentru cuvintele cheie și forma astfel titluri chiar din aceste cuvinte. Observați ce titlu va apărea pentru jocul PING dacă acesta se va salva după ce linia 3 este înlocuită cu:

```
   3 LET a$=a$+CHR$ 18+CHR$ 1+CH
R$ 19+CHR$ 1+CHR$ 22+CHR$ 1+CHR$
 0+CHR$ 219+CHR$ 239
```

3. Utilizarea variabilelor de sistem

Octeții de memorie de la adresa **23552** la adresa **23733** sînt rezervați pentru operații specifice sistemului. Ei pot fi citiți pentru a afla diverse lucruri despre sistem, iar unii dintre ei pot fi modificați. Acești octeți se numesc **variabile de sistem** și au cîte un nume. Unele variabile de sistem sînt localizate într-un singur octet de memorie, altele pe doi octeți de memorie și, mai puține, pe trei sau chiar mai mulți octeți. Fiind vorba de adrese de memorie, în cazul variabilelor formate din mai mulți octeți, primul (adresa cea mai mică) va fi octetul cel mai puțin semnificativ.

Modificarea valorilor pentru variabile de sistem conduce (poate) la cele mai spectaculoase rezultate pentru jocuri, rămînînd totuși în sfera BASIC-ului. Se pot astfel realiza artificii interesante și eficiente cu un efort minim. Dar atenție! modificarea unor variabile de sistem poate conduce la o funcționare eronată a sistemului; de aceea acest lucru nu este în general recomandat. De obicei, în tabelele în care se regăsesc aceste variabile se indică variabilele asupra cărora nu se poate acționa, precum și cele a căror modificare nu are efect asupra funcționării normale a sistemului.

Protecția jocurilor la întreruperi

De obicei, pentru astfel de protecții se folosește variabila de sistem **ERR SP** (la adresele de memorie **23613—23614** în care se pot introduce diverse valori), deoarece aceasta este variabila de sistem care specifică adresa de intoarcere în caz de eroare.

Astfel, dacă vom introduce valoarea **0** în cele două locații de memorie specificate, sistemul se va bloca atunci cînd se va încerca oprirea programului prin STOP. De exemplu, dacă se adaugă la jocul **Traversarea străzii** următoarea linie:

```
5 POKE 23613,0: POKE 23614,0
```

Încercarea de a opri programul cu STOP (SYMBOL SHIFT și A), cînd se va solicita introducerea unui număr pentru gradul de dificultate, se va solda cu blocarea sistemului. La alte valori, de exemplu pentru:

```
5 POKE 23614,244
```

sistemul va face NEW la STOP.

De remarcat că pentru alte jocuri (programe), la modificarea valorii variabilei **ERR SP** în modurile arătate, sistemul se va bloca sau va face NEW și în cazul unei tentative de a opri programul cu BREAK. Aceleași efecte se produc și atunci cînd programul se oprește datorită unei erori (de exemplu, se acționează o literă atunci cînd se așteaptă introducerea unei valori numerice pentru o variabilă numerică).

Cu această variabilă de sistem se poate realiza și protejarea programelor contra unor erori. De exemplu:

POKE 23613, PEEK 23730-5

va proteja programul contra oricărei erori, mai puțin cea de tip C, iar

POKE 23613, **PEEK** 23730-3
va reface posibilitatea BREAK-ului

Tot în scopul protejării programelor la întreruperi sau la **listări** se poate utiliza variabila de sistem **ELINE** (adresa **23641—23642**). Dacă această variabilă ia valoarea **0**, se poate întîmpla ca tastatura să se blocheze la comenzi sau să se distrugă programul. Astfel, în jocuri, se poate atribui variabilei valoarea **0** atunci cînd se va răspunde « nu » la întrebarea privitoare la dorința de a mai juca o dată. De exemplu, la VÎNĂTOAREA, dacă se modifică linia 6130 (care se va executa în cazul în care nu s-a răspuns prin « da » la întrebare) astfel:

```
6130 POKE 23642,0
```

Dacă se va încerca să se listeze programul după ce jocul s-a terminat, programul se va autodistruge.

Altă variabilă de sistem care poate fi folosită la protejarea programelor la întrerupere este **DF SZ** (adresa **23659**) care conține numărul liniilor din partea de jos a ecranului. Astfel, dacă se va introduce **0** în această locație de memorie (**POKE 23659,0**), acest lucru este echivalent cu declararea a zero linii în partea de jos a ecranului. În aceast caz, la o întrerupere a programului cu BREAK, programul se va distruge, sistemul nemaiavînd unde scrie mesajele. De reținut că valoarea variabilei se reface atunci cînd sistemul ajunge la o linie INPUT, iar cu **POKE 23659,1** linia 22 devine accesibilă cu PRINT.

Protecția jocurilor la încărcare (salvarea versiunilor protejate)

Uneori se dorește ca jocul (programul) să nu poată fi nici măcar citit (încărcat de alți utilizatori. Acest lucru se poate realiza, de exemplu, prin modificarea valorii variabilei de sistem **PROG** (adresa **23635, 23636**) care indică adresa programului BASIC.

Ştim că programul BASIC începe de la adresa **23755.** Dacă vom interoga variabila PROG (PRINT PEEK 23635) vom găsi 203, aceasta fiind valoarea din octetul cel mai puțin semnificativ pentru reprezentarea numărului 23755 ($23755 = 203 + 92 \times 256$).

Pentru salvarea unei versiuni protejate la încărcare se va proceda astfel:

POKE 23635, **<n>**: **SAVE** "**<nume>**" unde **n** poate fi un număr natural mai mic ca 203.

Pentru încărcarea programului respectiv:
POKE 2365, **<n>**: **LOAD** "" neuitînd ca după încărcare (înainte de lansarea în execuție) să se restabilească valoarea variabilei PROG:

POKE 23635,203

Dacă nu se va cunoaște care a fost valoarea lui **n** la salvarea programului, acesta nu se va putea încărca.

De exemplu: salvați jocul **Vrăjitorul portocaliu** modificînd în prealabil valoarea de la adresa 23635 cu una din următoarele valori: 1; 10; 200. Resetați sistemul și încercați să încărcați jocul astfel salvat. Veți observa că dacă nu veți urma pașii indicați (modificarea valorii de la adresa 23635 cu valoarea aleasă în prealabil 1; 10 sau 200, încărcarea programului și apoi readucerea valorii de la adresa 23635 la 203) nu veți mai reuși să încărcați corect jocul.

Prin modificarea variabilei **PROG** se poate obține și protejarea primei linii de program (linie 0) cu un rezultat similar cu cel obținut prin modificarea adresei de memorie 23755. În acest caz se va introduce:

POKE (PEEK 23635+256* PEEK 23636 +1),0

După cum se poate observa, avantajul constă în faptul că nu mai sînt necesare două POKE-uri, ci unul singur, nemaifiind necesară nici descompunerea numărului (nou) de linie, chiar dacă va fi un număr mai mare de 255.

Artificii necesare la jocuri cu dialog

Multe jocuri (chiar și din cele prezentate) se desfășoară sub forma unui dialog permanent între calculator (care de obicei pune întrebări) și jucător (care de obicei indică răspunsuri). Este în special cazul la jocurile didactice pentru formarea de-

prinderilor precum și la cele de aventuri. Cel puțin două probleme se pun pentru realizarea unui dialog « fără probleme » între calculator și jucător și care se pot soluționa ușor prin intermediul variabilelor de sistem.

Prima problemă se referă la faptul că atunci cînd ecranul se umple cu text (în urma dialogului creat) și se atinge ultima linie (linia 21), sistemul va afișa automat în partea de jos a ecranului mesajul « scroll? », ceea ce nu numai că poate crea confuzii pentru un jucător care nu cunoaște limbajul BASIC (și nu este neapărată nevoie de această cunoaștere pentru a se juca un joc pe calculator), dar și terminarea (nedorită) a jocului din cauza necunoașterii particularităților sistemului (prin acționarea uneia din tastele N sau BREAK).

Problema se poate soluționa prin modificarea variabilei de sistem **SCR CT** (adresa **23692**) care contorizează « scroll»-urile. Valoarea ei este totdeauna mai mare cu o unitate decît numărul de « scroll»-uri care vor fi făcute înaintea opririi prin mesajul « scroll?». În acest caz, dacă se va face **POKE 23692, 255,** se va obține un « scroll» permanent fără să mai apară mesajul. Acest artificiu s-a utilizat, de exemplu, în jocurile **Vrăjitorul portocaliu** (linia 290), **Comoara din peșteră** (liniile 5,100 și 300) și **Pierdut în junglă** (liniile 1 și 100).

A doua problemă poate apărea în cursul unui dialog, cînd șiruri mai lungi de caractere, reprezentînd răspunsuri, sînt memorate sau comparate cu alte șiruri de caractere, literele mici nefiind la fel interpretate ca literele mari. Deoarece răspunsurile pot fi și mai lungi (nu numai de tipul « da», « nu», « Da», « D» etc) este practic imposibil de a se analiza toate răspunsurile posibile reprezentînd combinații de litere mari și litere mici. Sigur, este posibil să se indice în instrucțiuni prin mesaje complete de genul: « Introduceți toate răspunsurile cu litere mari», dar această rezolvare (neelegantă) are cîteva dezavantaje: deturnarea atenției jucătorului către probleme auxiliare (nelegate direct de jocul propriu-zis), creșterea volumului de text memorat în program, neexcluderea posibilității de nerespectare a indicației de către jucător, uitarea acesteia etc. Problema se poate soluționa simplu prin modificarea variabilei de sistem **FLAGS2** (adresa **23658**). Dacă în cursul unui joc (program), în adresa de memorie pentru variabila de sistem **FLAGS2** se introduce valoarea **8 (POKE 23658,8),** atunci pe parcursul utilizării programului cursorului va fi în modul **C** (ca și cum s-ar fi acționat **CS+2**), iar dacă se introduce valoarea **16 (POKE 23658,16)** cursorul va fi în modul **L**. Prima versiune s-a utilizat în jocul **Pierdut în junglă** (linia 1), iar a doua în jocul **Animale** (linia 1010).

Un nou set de caractere

În realizarea unor jocuri, se poate dori ca afișarea textelor să se facă cu un set (complet) nou de caractere. Am văzut că setul de caractere este memorat în memoria ROM de la adresa 15360. Această informație se poate obține interogînd variabila de sistem **CHARS** (adresa **23606, 23607**), care va returna o valoare reprezentînd adresa de start a secvenței de caractere minus 256. Adresa primului caracter grafic se poate afla prin interogarea variabilei de sistem **UDG** (**23675, 23676**). Deoarece valoarea conținută în adresele 23606 și 23607 (variabila **CHARS**) este modificabilă, înseamnă că se pot obține alte valori de start pentru secvențe de caractere memorate în memoria RAM și astfel să se obțină un nou set de caractere pentru sistem utilizînd facilitatea de creare a caracterelor grafice definite de utilizator.

Cu variabila **CHARS** se pot obține și efecte interesante, după cum se va observa la rularea următorului program:

```
10 FOR I=0 TO 72
20 POKE 23606,I
30 PRINT 0: BEEP 0.01,I/5
40 NEXT I
50 POKE 23606,0
```

Dacă programul se va opri cu BREAK, nerefăcîndu-se valoarea din locația 23606, caracterele afișate pe ecran vor arăta, în continuare, foarte ciudat.

alte artificii pentru jocuri utilizînd variabilele de sistem

Deseori, în jocuri poate apărea parametrul timp, intervalul în care se obțin performanțele (atît în jocurile de reflexe cît și în cele logice), ajustînd rezultatele finale.

Într-un program (joc), măsurarea timpului scurs între două evenimente se poate realiza prin contorizarea acestuia într-o variabilă. De obicei, timpul scurs se măsoară printr-un ciclu FOR-NEXT, putîndu-se calcula aproximativ în funcție de valoarea la care a ajuns indicele ciclului. Dar, aceasta este o măsurare foarte aproximativă pentru un jucător care percepe timpul în secunde. În afară de o măsurare cît mai exactă a timpului, în jocuri, este necesară uneori afișarea unui ceas care să exprime timpul efectiv consumat în joc. **O măsurare mai exactă a timpului** cu calculatorul se poate realiza prin intermediul variabilei de sistem **FRAMES** (pe trei octeți la adresele **23672, 23673** și **23674**), care indică contorul de cadre. Știm că PAUSE 42, de exemplu, marchează aproximativ trecerea unei secunde. Citind valorile din locațiile variabilei de sistem **FRAMES** și anume 23674, 23673 și 23672 se pot memora incremente mai precise de 20 ms. Fiecare locație variază de la **0** la **255**, după care se reîncepe. Cel mai rapid se incrementează locația 23672 (cu 1 la fiecare 20 ms); cînd se trece de la **255** la **0**, locația 23673 se incrementează cu 1 și apoi analog pentru 23674. De exemplu, pentru a poziționa ceasul pe ora 15 și 30 de minute se procedează astfel:

POKE 23674,42: POKE 23673,146: POKE 23672,112 conform calculului $(15 \times 60 + 30)^{*} \times 60^{**} \times 50 = 2790000$ cinci zecimi de secundă, iar $2790000 = 65536 \times 42 + 256 \times 146 + 112$.

Ceasul intern al calculatorului poate fi foarte util în unele jocuri pentru cronometrarea jucătorului (timpul scurs pentru ieșirea dintr-un labirint, timpul acordat efectuării unei mutări într-un joc logic etc.).

Atenție! Instrucțiunile **BEEP, SAVE** și **LOAD** au ca efect «oprirea» ceasului (acesta este dezarmat pe timpul execuției instrucțiunilor respective). Astfel o măsură precisă a timpului și generarea de sunete într-un joc sînt în general incompatibile.

Prezentăm în continuare un program joc care pune în funcțiune ceasul intern al calculatorului pentru a măsura timpul scurs între afișarea unui caracter pe ecran și acționarea tastei corespunzătoare, această tehnică putînd fi aplicată, în general, în jocurile care necesită măsurarea timpului scurs între două evenimente:

* = minute
** = secunde

```
  5 REM REFLEXE
 10 BORDER 0: PAPER 0: INK 4: C
LS
 20 DEF FN t()=INT ((65536*PEEK
 23674+256*PEEK 23673+PEEK 23672
)*0.2)
 30 LET i=INT (RND*89+33): IF i
=91 OR i=92 OR i=93 THEN GO TO 3
0
 40 LET a$=CHR$ i
 50 LET X=INT (RND*32): LET Y=I
NT (RND*20)
 60 FOR i=0 TO 2
 70 POKE 23672+i,0
 80 NEXT i
 90 PRINT AT Y,X;a$
100 LET b$=INKEY$: IF b$<>a$ TH
EN GO TO 100
110 PRINT AT 21,0;"TIMPUL: ";FN
 t();" zecimi de secunda"
130 PAUSE 0
140 PAUSE 0
150 RUN
```

În linia 20 se definește o funcție care reprezintă numărul de zecimi de secundă scurs de la ultima punere la zero a ceasului. În liniile 30 și 40 se alege un caracter cu eliminarea anumitor caractere accesibile numai în modul extins.
În liniile 60—80 se pune la zero ceasul.

Variabila de sistem **MODE** (adresa **23617**) specifică cursorul (**K**, **L**, **C**, **E**, **G**). Modificînd această variabilă se pot obține **diverse caractere pentru cursor**, inclusiv un caracter grafic.
Exemplu de obținere a unui cursor « ciudat»: **POKE 23617,30**

Deoarece programele joc pot căpăta dimensiuni destul de mari, este important ca liniile de program să fie editate cu ușurință.
Variabila de sistem **REPPER** (adresa **23562**) determină timpul necesar acționării unei taste, astfel încît aceasta să se repete. Inițial, valoarea variabilei REPPER este 35, dar dacă ea se modifică, de exemplu:

POKE 23562,1

atunci tastatura va fi citită mult mai repede, ceea ce dă posibilitatea unei editări mult mai eficiente a liniilor de program mai lungi care necesită modificări (cursorul se deplasează mult mai rapid). Modificarea se va face în mod comandă, înainte de a se începe editarea unor linii de program.

Variabila de sistem **PIP** (adresa **23609**) conține durata sunetului la apăsarea unei taste. Folosindu-se în jocuri, de exemplu **POKE 23609,5** sau **POKE 23609,10** se va obține un sunet caracteristic (clic) la acționarea unei taste. Cu cît numărul introdus la adresa 23609 va fi mai mic, cu atît sunetul va fi mai scurt. Pentru 255 se obține sunetul cu cea mai lungă durată. În jocuri, artificiul este eficient permițînd jucătorului controlul acționării unei taste. De asemenea, poate ajuta și cînd se efectuează o editare a liniilor unui program.

Variabila de sistem **REPDEL (23561)** arată timpul (în 1/50-imi de secundă) cît timp trebuie să fie acționată o tastă pentru a fi citită (în mod normal 35), iar **REPPER (23562)** arată timpul necesar pentru a reciti o tastă ținută apăsată (în mod normal este 5). Modificînd aceste variabile se poate schimba viteza de citire a tastelor cu INPUT.

Variabila **RASP (23608)** indică lungimea sunetului de avertizare (eroare). În mod normal, valoarea variabilei este 64, iar această valoare se poate modifica.

Variabila **BORDCR (23624)** arată culoarea BORDER × 8 + atributele caracteristice părții de jos a ecranului (liniile 22 și 23). De exemplu, cu **POKE 23624,200**, mesajele de pe liniile din partea de jos a ecranului se vor vedea foarte greu. **POKE 23624,0** va face atributele celor două linii de jos **0** iar cursorul de editare nu se va mai vedea deloc.

Variabila **EPPC (23625, 23626)** indică numărul liniei curente (pe care se află cursorul). Dacă valoarea din această locație se modifică, se poate puncta o anumită linie gata pentru editare. De exemplu, cu: **POKE 23625,10**: **POKE 23626,0** cursorul se va muta la linia cu numărul **10**.

PEŞTERA

LEGENDA;
Numere — numărul încăperii sau pași de program
Numere încercuite — locul unde se ascunde comoara (1 — la primul joc, 2 — la al doilea etc.)
0 — parter
1 — etaj
—1 — subsol

⊙ pe aici se poate urca
⊗ pe aici se poate coborî (E_0 corespunde lui E_1, A_0 lui A_1 și A_{-1} la urcare și coborîre)
- - - - - tunel prea îngust — nu se poate trece cu comoara, care se află într-o ladă de dimensiuni mari
B—B — pe aici se plimbă (din loc în loc) Bill Bones
⇑ — pe aici se poate ieși din peșteră

Bibliografie

1. D. AHL, **100 Computer Games,** 1981.
2. X. L. BELLEFONDS, **La pratique de ZX Spectrum: Basic aprofondi** Editions du PSI, Paris, 1983.
3. J. BERNARD, **50 programmes ZX Spectrum** Editions Radio, Paris, 1983.
4. E. R. BERLEKAMP, J. H. CONWAY, R. K. GUY, **Winning Ways for your Mathematical Plays,** Academic Press, New Yord, London, 1982.
5. I. DIAMANDI, **Partenerul meu de joc — calculatorul,** RECOOP, București, 1988.
6. I. DIAMANDI, C. CĂLINESCU, **Dialog cu viitorul,** Ed. Științifică și Enciclopedică, București, 1988.
7. Gh. FEȚEANU, Gh. PĂUN (coord.), **Cartea jocurilor,** RECOOP, București, 1988.
8. M. GARDNER, **Amuzamente matematice,** Ed. Științifică, București, 1969.
9. M. GARDNER, **Alte amuzamente matematice,** Ed. Științifică, București, 1970.
10. Gh. PĂUN, **Între matematică și jocuri,** Ed. Albatros, București, 1986.
11. Gh. PĂUN, **Soluții pentru 50 de jocuri logice solitare,** RECOOP, București, 1987.
12. A. PETRESCU și colab., **Totul despre calculatorul personal aMIC,** Ed. Tehnică, București, 1985.
13. A. STOICA, I. DIAMANDI ș.a., **Elemente de informatică pentru cercurile tehnico-științifice ale elevilor,** U.T.C., București, 1988.
14. S. WICKERS, **Sinclair Spectrum BASIC Programming,** Sinclair Ltd., Cambridge, 1982.
15. xxx **Perspectives (Revue trimestrielle de l'education),** UNESCO, 27, 3 (1987): Dossier — L'informatique dans l'education.

- **Redactori:** MIHAELA IONESCU și EMILIA TEODORU
- **Coperta 1:** PAULA VLĂDESCU
- **Coperta interioară și schițe:** EMIL BOJIN
- **Prezentare grafică și artistică:** PETRE POPESCU

EDUCATIVE
JECO
COLECTIVE

LEI 150